

This chapter describes how you can develop printer drivers for printing documents with QuickDraw GX on printing devices. For example, you might develop a printer driver that allows a third-party printer or plotter to print documents created by any application that uses QuickDraw GX.

To use this chapter, you need to be familiar with how the printing features in QuickDraw GX work. *Inside Macintosh: QuickDraw GX Printing* describes these printing features and gives you an overview of the printing process.

Printer drivers include a number of resources. Before reading this chapter, you need some understanding of Macintosh resources and resource files. You can read about them in the Resource Manager chapter in *Inside Macintosh: More Macintosh Toolbox*.

Printer drivers use collections, which are managed by the Collection Manager. Drivers are activated by messages, which are managed by the Message Manager. You need to know about the Collection Manager and Message Manager components of QuickDraw GX to understand how to develop drivers. The chapter “Introduction to Printing Extensions and Drivers” in this book provides an overview of using these managers in printer drivers. Both of them are described in *Inside Macintosh: QuickDraw GX Environment and Utilities*.

This chapter begins with an overview of printer drivers and then discusses

- the tasks that any printer driver must accomplish
- the source files for a typical printer driver
- the printing messages used to create a printer driver
- the ImageWriter II printer driver as an example of how to construct a printer driver
- the resources used to create the ImageWriter II driver
- the user interface requirements for QuickDraw GX printer drivers

For specific reference information about the messages that you override to develop a printer driver, see the chapter “Printing Messages” in this book. For specific reference information about the resources that you need to include in your printer driver file, see the chapter “Printing Resources” in this book.

#### Note

The code samples presented in this chapter are taken from a recent version of the the sample code. These samples are not complete and may have been slightly modified since printing. You can find the latest version of the code in the QuickDraw GX sample code. ♦

## About Printer Drivers

Macintosh system software uses device drivers to communicate information to hardware devices. A **printer driver** is a device driver—an independent software component that the system software uses to convert document data into printed output on a peripheral device such as a printer or plotter. In the QuickDraw GX environment, a printer driver

## Printer Drivers

sends QuickDraw GX data and instructions in a form specific to the printing device that it drives and manages the communications with that printing device.

Each printer driver has a Chooser interface through which the user can select communications parameters or special features of the driver. Each driver also displays status and alert information to the user through the Finder, as described in the section “Displaying Status Information and the Printing Alert Boxes” beginning on page 3-41.

QuickDraw GX allows the user to print documents to a different computer than is used to create the documents. This is often the case when a network includes a print server that several computers share. When the same computer is used, only one copy of your driver is required. When different computers are used, a copy of your printer driver must reside both on the computer that is sending the document to be printed and on the computer that is communicating with the printing device. The use of more than one copy of your driver is generally invisible to you, but understanding it does come into play because of how status and error conditions are handled in your driver. This is described in the section “Handling Status and Alert Conditions” beginning on page 3-9.

You need to develop a separate driver for each hardware device that has different characteristics. Developing a printer driver is different for each imaging system.

- **Raster imaging system** drivers convert QuickDraw GX shapes into raster data—blocks of data representing the pixels in an image—which are sent to raster printing devices to produce printed output. The Apple ImageWriter is an example of a raster printing device.
- **PostScript imaging system** drivers convert QuickDraw GX shapes into PostScript instructions, which are sent to PostScript printing devices to produce printed output. The Apple LaserWriter Pro 600 is an example of a PostScript printing device.
- **Vector imaging system** drivers convert QuickDraw GX shapes into vectors, which are sent to vector printing devices to produce printed output. Plotters that use HPGL are examples of vector printing devices.

A printer driver consists of resources that define the driver to the printing system. A printer driver is a message handler in a message chain. QuickDraw GX communicates with a driver by sending printing messages down the message chain. A driver receives a printing message by defining the set of messages it wishes to receive and act on. For example, a printer driver that supports raster printing devices overrides the messages for the raster imaging system.

Although you need to specify a fair amount of information in your printer driver resources, you can create drivers for many printing devices without writing very much code. QuickDraw GX provides default implementations of all of the tasks that a printer driver needs to accomplish, which means that you only need to override the printing messages that relate to specific or unique qualities of your printing device. For example, some PostScript printer drivers can use all of the default implementations of printing messages, with any device-specific parameters defined in the resource descriptions that you provide.

Each of the following sections describes a task that a QuickDraw GX printer driver needs to manage:

- writing document data to the spool-file when the application initiates the printing process
- reading document data from the spool file when creating the printed output
- converting QuickDraw GX shapes into a format that the printing device can understand
- handling the physical communication of data with the printing device
- interacting with the Finder to respond to Printing menu events
- providing a Chooser interface so that users can select your driver
- reporting status and alert conditions to the user
- providing compatibility with the Macintosh Printing Manager application interface

Each printing task is accomplished during one of the phases of printing, which are described in the chapter “Introduction to Printing Extensions and Drivers” in this book. Some of the printing tasks execute in the foreground and others execute in the background. A **foreground task** takes control of the computer, preempting the user from performing any other application tasks. A **background task** operates by taking control of the computer for brief periods of time while the application has relinquished time, allowing the user to continue working.

## Writing Data to the Spool File

The first task that a QuickDraw GX printer driver needs to manage involves spooling a document to disk. During spooling, which occurs in the foreground, the application writes document data to a spool file by sending an intermediate representation of the document (which can be a portable digital document) to disk. Spooling occurs when the application calls the `GXStartJob` function, which causes QuickDraw GX to send the `GXStartJob` message.

Spooling of the document data to the spool file is handled by five messages: `GXCreateSpoolFile`, `GXSpoolPage`, `GXSpoolData`, `GXSpoolResource`, and `GXCompleteSpoolFile`.

The QuickDraw GX default implementations of the spooling messages provide the basic spooling functionality. You generally override these messages in a printing extension or printer driver to add specific data handling, such as encrypting the data that is sent to the spool file. If you are taking advantage of the default handling of these spooling messages, you always need to forward the messages before or after adding your own functionality.

If you choose to totally override a spooling message, that is, you choose not to forward it so that the default implementation can perform its tasks, you must handle all spooling on your own. This means that you must totally override all of the spooling messages and take care of writing the data to the spool file. You only need to do this if you are developing a driver that needs to create a custom spool-file format.

## Printer Drivers

You can find a complete description of the interface and functionality of each of the spooling messages in the section “Spooling Messages” beginning on page 4-67 in the chapter “Printing Messages.”

### Spooling Translated QuickDraw Data in Print Files

When spooling a file that contains QuickDraw drawing commands, QuickDraw GX by default stores untranslated QuickDraw data in the print file. When the print file is despoiled and imaged, QuickDraw GX then converts the data to QuickDraw GX shapes and prints them. If you want to force QuickDraw GX to translate QuickDraw data to QuickDraw GX shapes before spooling, you can override the `GXFetchTaggedData` message that QuickDraw GX sends at spooling time to fetch resource data of type `'pcfg'`. Your override should forward the message, and then clear the highest-order bit of the data that is returned from the message. If the data's highest-order bit is cleared, QuickDraw GX translates and stores the data in the print file as QuickDraw GX shapes. For more information on QuickDraw data in print files, see the discussion of the `'pict'` tag object in the advanced printing features chapter of *Inside Macintosh: QuickDraw GX Printing*. ♦

### Reading Data From the Spool File

---

During the first part of the imaging phase of printing, QuickDraw GX despoils each document from disk. During despooling, which occurs in the background, your printer driver reads the spooled document from the spool-file. Depooling of the document data from the spool-file is handled by five messages, which are complementary to the spooling messages: `GXCountPages`, `GXDespoolPage`, `GXDespoolData`, `GXDespoolResource`, and `GXCloseSpoolFile`.

The QuickDraw GX default implementations of the despooling messages provide the basic despooling functionality. You generally override these messages in a printing extension or printer driver to add specific data handling, such as decrypting the data that you encrypted in your overrides of the spooling messages. If you are taking advantage of the default handling of these despooling messages, you always need to forward the messages before or after adding your own functionality.

If you choose to totally override a despooling message, that is, you choose not to forward it so that the default implementation can perform its tasks, you must handle all despooling on your own. This means that you must totally override all of the despooling messages and take care of reading the data from the spool-file. As with spooling, you only need to do this if you are implementing a driver that needs to create a custom spool-file format.

You can find a complete description of the interface and functionality of each of the despooling messages in the section “Despooling Messages” beginning on page 4-74 in the chapter “Printing Messages.”

### Despooling Print Files Containing QuickDraw Picture Data

When a document containing QuickDraw imaging commands is spooled, QuickDraw GX by default saves the QuickDraw data for each page in a tag object of tag type 'pict' attached to a rectangle shape. If you examine the contents of a despoiled file, note that the presence of a rectangle shape with such an attached tag object indicates the presence of QuickDraw picture data. For more information, see the discussion of the 'pict' tag object in the advanced printing features chapter of *Inside Macintosh: QuickDraw GX Printing*. ♦

### Converting QuickDraw GX Shapes

During the second part of the imaging phase of printing, your printer driver needs to convert the QuickDraw GX shape or shapes that compose each page into instructions that your printing device can use to produce the printed version of the page. This process of rendering each page into printable form takes place in the background. Different kinds of printing messages are sent during rendering:

- Universal imaging messages are always sent during the rendering phase. The universal imaging messages include GXSetupImageData, GXImageJob, GXImageDocument, GXImagePage, and GXRenderPage.
- Raster imaging messages are only sent when a user is printing to a device that uses the raster imaging system. The raster imaging messages are: GXRasterDataIn, GXRasterLineFeed, and GXRasterPackageBitmap.
- PostScript imaging messages are only sent when a user is printing to a device that uses the PostScript imaging system. Some of the PostScript imaging messages are: GXPostScriptQueryPrinter, GXPostScriptGetPrinterGlyphsInformation, GXPostScriptEjectPage, and GXPostScriptProcessShape.
- Vector imaging messages are only sent when a user is printing to a device that uses the vector imaging system. The vector imaging messages are: GXVectorPackageShape, GXVectorLoadPens, and GXVectorVectorizeShape.

The QuickDraw GX default implementations of the imaging messages provide the basic imaging functionality. You generally override some of these messages to manage the specifics of your device. Although you can override any of the messages to customize your driver, you often only need to override the GXSetupImageData universal imaging message to initialize your device, and then override some of the messages specific to the imaging system to handle your device.

If you are developing a printer driver for a PostScript device, you can likely use the default implementations for many PostScript devices, whereas you usually have to override the raster and vector imaging messages to generate the unique data sequences required by each raster and vector device.

You can find a complete description of the interface and functionality of each of the imaging messages in the chapter “Printing Messages” in this book.

## Communicating With the Printing Device

---

During the device communications phase of printing, QuickDraw GX sends the data that composes the rendered form of each page to the printing device. This phase takes place in the background and is the phase during which physical communications with your device takes place.

### Note

The device communications phase of printing can take place on a different computer than the computer on which imaging takes place. You need to keep this in mind when writing your code. For example, the code that you use to access a file during this phase needs to take this fact into account. ♦

QuickDraw GX normally handles device communications with these operations:

- Initiates communications with the `GXOpenConnection` message and terminates communications with the `GXCloseConnection` message.
- Initiates the sending of data for a page with the `GXStartSendPage` message and terminates sending of the page data with the `GXFinishSendPage` message.
- Adds data for a page to an output buffer with the `GXBufferData` message. QuickDraw GX sends the buffered data to the printing device with the `GXDumpBuffer` message. QuickDraw GX then sends the `GXFreeBuffer` message to wait for the completion of buffer processing.
- Sends data directly to the printing device (without buffering) with the `GXWriteData` message.

The QuickDraw GX default implementations of the device communications messages handle communicating with your printing device for you. QuickDraw GX provides different versions of these messages for the different kinds of standard I/O: serial, Printer Access Protocol (PAP), or not-connected. You specify the communications type of your printing device in your driver resources, as described in the chapter “Printing Resources” in this book.

You override the printing device communications messages when you need to customize the handling of the communication. For example, you can override the `GXOpenConnection` message to verify that a printing device of your type is connected to the computer and that the printing device is working properly. If you are writing a driver to capture printer output and store it in a file rather than directly communicate it to a printing device, you can override the `GXBufferData` message and store the buffered data in a file. If you need to modify the primitive I/O handling for your printing device, you can override the `GXWriteData` message.

You usually forward these messages after adding your own instructions about handling them. If you choose to create your own communications buffering scheme, you can totally override the `GXBufferData`, `GXDumpBuffer`, and `GXFreeBuffer` messages.

You can find a complete description of the interface and functionality of each of the printing device communications messages in the section “Device Communications Messages” beginning on page 4-131 in the chapter “Printing Messages.”

## Interfacing With the Finder

---

The Macintosh Finder sends five messages that you might need to override in your driver: `GXGetDTPMenuList`, `GXDTPMenuSelect`, `GXInitializeStatusAlert`, `GXHandleAlertEvent`, and `GXHandleAlertFilter`. You override these messages to add any items to the Printing menu of the Finder and to handle events that occur in those items.

You can find a complete description of the interface and functionality of each of these Printing menu messages in the section “Finder Menu Messages” beginning on page 4-169 in the chapter “Printing Messages.” Desktop printers and the menu choices associated with them are described in *Inside Macintosh: QuickDraw GX Printing*.

## Providing a Chooser Interface

---

You need to make it possible for the user to choose your driver in the Chooser and to set any options that you provide, including various possibilities for communicating with your printing device, such as port options and baud rates.

You specify the connection possibilities for your driver in a look (`'look'`) resource, which is used by the Chooser to display choices to the user. The look resource references the communications (`'comm'`) resources that you provide, which allows QuickDraw GX to configure the device communications after the user makes a selection. The look and communications resources are discussed in the section “Using Resources in Drivers” beginning on page 3-53 and are described in detail in the chapter “Printing Resources” in this book.

You also need to define Chooser package (`'PACK'`) and list definition (`'LDEF'`) resources for the Finder interface to your printer driver, just as you do for any other Macintosh driver. These resources are described in the Finder interface chapter in *Inside Macintosh: More Macintosh Toolbox*.

## Handling Status and Alert Conditions

---

During the device communications phase of printing, you need to provide status information to the user through the Finder, and you need to handle error or alert conditions that occur on your printing device.

Status information is displayed in the desktop printer window, which is provided by the Finder. While your printer driver is handling the printing of a job, you update the status information that is displayed in this window with text strings such as “Downloading fonts to printer” or “Opening connection.”

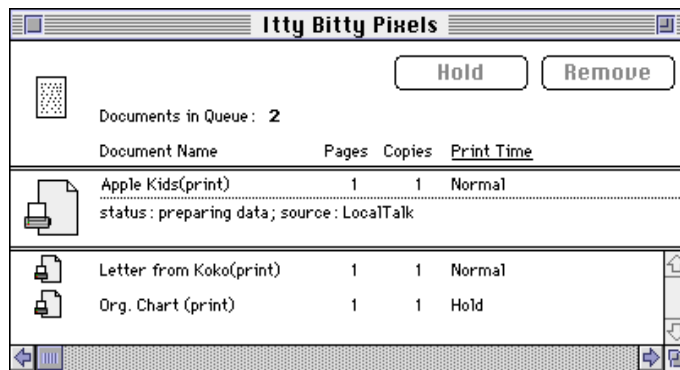
When a user is printing on a network with a print server, two copies of your driver are in use at the same time: one on the computer on which the user initiated printing (the client), and another on the server that is connected to the printing device. Several user machines can be communicating with the print server simultaneously, so the Finder arbitrates which status information is sent to which driver on which user machine.

In status (`'stat'`) resources that you include with your driver, you define the status text. When you want to display status information to the user, you call the

## Printer Drivers

`GXReportStatus` function, including the ID of the status text that you want shown. The Finder receives the status information from the `GXReportStatus` function and sends a `GXWriteStatusToDTPWindow` message to the copy of your driver that is running on the machine that initiated printing. The default implementation of the `GXWriteStatusToDTPWindow` message converts the status text ID into a string (by accessing the resource) and calls the `GXWriteDTPData` function to display the string. Figure 3-1 shows the display of a status text string in the desktop printer window.

**Figure 3-1** The printing status displayed in a desktop printer window



You can override the `GXWriteStatusToDTPWindow` message if you need to add information to the status text string at run time. For example, if you are composing a status text string that includes the current page number, you could override this message. In your override, you would determine if the current status text string is the one that you want to change, and produce a text string that incorporates the page number into it.

You need to alert the user when certain conditions arise during the printing of a document; for example, when the printing device runs out of paper, when the printing device requires that a sheet of paper be fed manually, or when a hardware error has occurred.

When these kinds of conditions occur, you have to alert the user, who then must take an action before printing can continue. The standard way to handle this is with a printing alert box. QuickDraw GX can manage displaying your printing alert box when you define printing alert ('plrt') resources. These resources automate the creation and display of alert boxes.

You implement handling of a printing alert box with the `GXAlertTheUser` function, as described in the chapter "Printing Functions for Message Overrides" in this book. An example of using the `GXAlertTheUser` function is shown in Listing 3-15 on page 3-42.



You can find a complete description of the interface and functionality of each of the status and error-handling messages in the chapter “Printing Messages” in this book. You can find descriptions of the status and printing alert resources in the chapter “Printing Resources” in this book.

### Providing Compatibility With the Macintosh Printing Manager

---

A final task to manage in your printer driver is how to interface to applications that use the Macintosh Printing Manager in the Macintosh Operating System. QuickDraw GX provides a default translation from Macintosh Printing Manager calls into QuickDraw GX messages, which allows those applications to use your printer driver without change.

You can customize this conversion with several resources. You can also override the default implementations of the Macintosh Printing Manager compatibility messages, which include such messages as `GXPrOpenDoc`, `GXPrCloseDoc`, `GXPrOpenPage`, `GXPrJobInit`, `GXPrValidate`, and `GXPrJobMerge`.

You can find a complete description of the interface and functionality of each of the Printing Manager compatibility messages in the section “Compatibility Messages” beginning on page 4-147 in the chapter “Printing Messages.” You can find a description of the compatibility resources in the section “Resources Used for Printing Extensions and Printer Drivers” beginning on page 6-12 in the chapter “Printing Resources.”

## Writing Printer Driver Files

---

After you know which messages you need to override as part of managing certain printing tasks, you need to write your printer driver in three kinds of files:

- You need to write a jump table in an assembly-language file so that QuickDraw GX can invoke the correct code in response to the printing messages that it sends.
- You need to write your message override functions in code files (typically, C language files).
- You need to define in Macintosh resource files the resources that contain the data for your printer driver.

The content of each of these file types is reviewed in this section, with sample code from the ImageWriter II printer driver included. The complete contents of the ImageWriter II driver files are found in the QuickDraw GX sample code.

Each driver must include one or more assembly-language jump tables, one or more files of code (in assembly-language, C, Pascal, or some other language) to write the functions

## Printer Drivers

that the driver performs, and one or more resource files. The ImageWriter II printer driver consists of the files shown in Table 3-1.

**Table 3-1** Files used to implement the ImageWriter II printer driver

Filename	Contents
<code>CommonDefines.h</code>	The header file containing values and types used in other files
<code>ChooserSupport.a</code>	The assembly-language defines for working with the Chooser-related resources
<code>ChooserSupport.c</code>	The code for the resources used by the Chooser
<code>ChooserSupport.r</code>	The resources needed for the driver to work with the Chooser
<code>newapp.a</code>	The jump table for the overrides provided in the <code>newapp.c</code> module
<code>newapp.c</code>	The implementations of the QuickDraw GX printing messages that this driver overrides
<code>newapp.r</code>	The resources needed for the QuickDraw GX printing messages
<code>oldapp.a</code>	The jump table for the overrides provided in the <code>oldapp.c</code> module
<code>oldapp.c</code>	The implementations of message overrides for compatibility with the Macintosh Printing Manager
<code>oldapp.r</code>	The resources needed for the Macintosh Printing Manager compatibility messages

This section describes the contents of several of these files. The complete files are found in the QuickDraw GX sample code.

The ImageWriter II printer driver is divided into two portions:

- The portion that provides the overrides of the QuickDraw GX printing messages is found in the files `newapp.a`, `newapp.c`, and `newapp.r`.
- The portion that provides the overrides of the Macintosh Printing Manager compatibility messages is found in the files `oldapp.a`, `oldapp.c`, and `oldapp.r`.

You can develop your printer driver in any number of files. Some developers find it convenient to separate the Macintosh Printing Manager compatibility portion of the code from the QuickDraw GX printing message portion, while others use different criteria to decide which code goes in which file.

## Message Overrides and the Jump Table

You need to tell QuickDraw GX where (in which segment and at what offset from the beginning of that segment) to find the code for each printing message that you are

overriding. You do this by defining an assembly-language jump table. The contents of the file `newapp.a`, which defines the jump table for the ImageWriter II printer driver, are shown in the QuickDraw GX sample code.

The jump table links the appropriate function into your code and provides a jump statement to invoke that function. The override ('over') resource tells QuickDraw GX where to look (at which offset) in the jump table to find the jump statement for a specific printing message name. In this way, QuickDraw GX knows which of your functions to call to respond to the messages in which you are interested.

You define a jump table with one jump (JMP) statement for each message that you override. You also define an override resource that specifies the offset in that jump table for each message. The jump table and override resource must be coordinated. QuickDraw GX dispatches a printing message to your printer driver by jumping to the routine whose location is specified in the appropriate location in the jump table. The override resource is described in the section “The Override ('over') Resource” beginning on page 6-13 in the chapter “Printing Resources.”

QuickDraw GX reserves the first 4 bytes for its own use, so the value of the constant `firstOffset` is 4. These first 4 bytes are used by QuickDraw GX to maintain an owner count and must be set to 0 in the jump table.

For example, the ImageWriter II printer driver overrides the messages that are shown in Table 3-2. This driver also overrides several Macintosh Printing Manager compatibility messages, which are not shown in this table.

**Table 3-2**      Printing messages overridden by the ImageWriter II printer driver

Message	Why you override
GXInitialize	To set up the environment so that the printer driver can use predefined global values
GXShutDown	To free the storage that was allocated by the GXInitialize function
GXDefaultPrinter	To modify the default printer object characteristics, such as adding view devices that the application can use
GXDefaultJob	To modify the default job characteristics, such as which format is the default format
GXJobDefaultFormatDialog	To modify the behavior or appearance of the Page Setup dialog box
GXJobFormatModeQuery	To provide support for direct-mode printing
GXOpenConnection	To establish connection with the printing device
GXSetupImageData	To modify the initialization data that is used for imaging the entire job

*continued*

## Printer Drivers

**Table 3-2** Printing messages overridden by the ImageWriter II printer driver (continued)

Message	Why you override
GXStartSendPage	To perform any actions required to set up printing for the next page, such as alerting the user to manually feed a piece of paper
GXRenderPage	To perform tasks during the imaging of each page
GXRasterPackageBitmap	To perform your own translation from bitmaps into character sequences for your device
GXRasterLineFeed	To send the codes to the printing device that cause it to move the print head

The override resources for the ImageWriter II printer driver must include an entry for each of these printing messages, and the jump table must include a jump statement for each. Listing 3-1 shows the override resource definitions from the `newapp.r` file. The override resource definition for the Macintosh Printing Manager compatibility message overrides, from the `oldapp.r` file, is shown in the QuickDraw GX sample code.

**Listing 3-1** Two override resources from the ImageWriter II printer driver

```
#define firstOffset 4
#define segmentID NewSegID

resource gxOverrideType (gxDriverUniversalOverrideID, sysHeap,
                        purgeable)
{
    {
        gxInitialize,           segmentID, firstOffset + 0,
        gxShutDown,             segmentID, firstOffset + 4,
        gxDefaultPrinter,       segmentID, firstOffset + 8,
        gxDefaultFormat,        segmentID, firstOffset + 12,
        gxDefaultJob,            segmentID, firstOffset + 16,
        gxJobDefaultFormatDialog, segmentID, firstOffset + 20,
        gxJobFormatModeQuery,    segmentID, firstOffset + 24,
        gxRenderPage,           segmentID, firstOffset + 28,
        gxOpenConnection,        segmentID, firstOffset + 32,
        gxCloseConnection,       segmentID, firstOffset + 36,
        gxStartSendPage,         segmentID, firstOffset + 40,
        gxSetupImageData,        segmentID, firstOffset + 44,
    };
};

/*
```

## Printer Drivers

The following are overrides for raster-specific messages, including where to find them in the jump table.

```
*/
resource gxOverrideType (gxDriverImagingOverrideID, sysHeap)
{
    {
        gxRasterPackageBitmap, segmentID, firstOffset + 48,
        gxRasterLineFeed,      segmentID, firstOffset + 52,
    };
};
```

In Listing 3-1, the name of each message is followed by the ID of the segment in which its code resides (in this case, the constant `segmentID`) and the byte offset of its jump statement in the jump table. QuickDraw GX reserves the first 4 bytes for its own use, so the value of the constant `firstOffset` is 4. These first 4 bytes are used by QuickDraw GX to maintain an owner count and must be set to 0 in the jump table.

The ImageWriter II printer driver includes two override resources for these printing messages, as shown in Listing 3-1: the first defines which universal printing messages the driver overrides, and the second defines which printing messages the driver overrides that are specific to the raster imaging system. You need to define the universal message overrides in a resource with a different ID than the resource in which you define the message overrides that are specific to the imaging system. This is described in the chapter “Printing Resources” in this book. The ImageWriter II driver also includes a third override resource for the Printing Manager compatibility messages that it overrides.

You implement the jump table as an assembly-language program that contains a jump statement for each override function. You need to list the jump statements in exactly the same order as you listed the message names in your override resource; otherwise, QuickDraw GX will invoke the wrong function in response to a message. Listing 3-2 shows the jump table for the QuickDraw GX messages found in the override resources in the `newapp.r` file.

**Listing 3-2** The jump table for the ImageWriter II printer driver

```
SD_JumpTable PROC EXPORT

                DC.L 0

                ; Universal messages
                IMPORT SD_Initialize
                JMP     SD_Initialize

                IMPORT SD_ShutDown
                JMP     SD_ShutDown
```

## Printer Drivers

```

IMPORT SD_DefaultPrinter
JMP     SD_DefaultPrinter

IMPORT SD_DefaultFormat
JMP     SD_DefaultFormat

IMPORT SD_DefaultJob
JMP     SD_DefaultJob

IMPORT SD_JobDefaultFormatDialog
JMP     SD_JobDefaultFormatDialog

IMPORT SD_JobFormatModeQuery
JMP     SD_JobFormatModeQuery

IMPORT SD_RenderPage
JMP     SD_RenderPage

IMPORT SD_OpenConnection
JMP     SD_OpenConnection

IMPORT SD_CloseConnection
JMP     SD_CloseConnection

IMPORT SD_StartSendPage
JMP     SD_StartSendPage

IMPORT SD_SetupImageData
JMP     SD_SetupImageData

; Raster messages
IMPORT SD_PackageBitmap
JMP     SD_PackageBitmap

IMPORT SD_LineFeed
JMP     SD_LineFeed

END

```

**Note**

The code shown in Listing 3-2 is for the MPW environment. If you are programming in a different development environment, you might need to use different assembler directives. You must, however, be certain to include the initial 4 bytes (set to 0) and each `JMP` statement. ♦

The `EXPORT` statement at the beginning makes the jump table public. The `IMPORT` statements make it possible for your assembly-language program to reference the C language functions that are performing your message overrides and to provide correct linkage to those functions.

Because QuickDraw GX uses the first 4 bytes in the jump table to store the owner count value, the first statement (`DC.L`) must be included. These bytes must all have the value 0 in them.

The name of each override function provided by the ImageWriter II printer driver is prefixed with the string `SD_` to differentiate it from other overrides of the same message. This means that the driver's override of the `GXInitialize` message is named `SD_Initialize`, its override of the `GXOpenConnection` message is named `SD_OpenConnection`, and so on.

You can choose to intersperse the `IMPORT` and `JMP` statements as shown in Listing 3-2, or you can place all of the `IMPORT` statements together, followed by all of the `JMP` statements, as is shown in Listing 2-2 on page 2-10 in the chapter “Printing Extensions.”

#### IMPORTANT

Always coordinate the entries in your override resources with the entries in your jump table. If they are not aligned, the wrong code will be executed for a message. The offset that you specify in the resource for each message must match the offset of the corresponding function in your jump table. You must also include 4 bytes with zero values at the beginning of your jump table. ▲

## The Message Overrides

The code that makes up your printer driver is a set of functions that override, either partially or totally, some of the printing messages that QuickDraw GX sends during the process of printing a document. This section describes how you use printing messages to develop your driver. It also describes how to use the printing collections to access and modify printing information, and how to use the `nrequire` macro to handle exceptions in your code.

Whenever you override a QuickDraw GX printing message, you must be certain that the declaration of your override function matches the declaration of the message. This means that the type of function return and the type of each parameter must match the types in the message declaration. The chapter “Printing Messages” shows the declaration of each printing messages.

QuickDraw GX provides a default implementation of each printing message that it sends. You can augment (partially override) some messages, and you can replace (totally override) others. For each printing message, QuickDraw GX provides one of three kinds of default implementations, as shown in Table 2-3 on page 2-12 in the chapter “Printing Extensions.”

The contents of the file `newapp.c`, which contains the source code for the QuickDraw GX message overrides in the ImageWriter II printer driver, are shown in the QuickDraw GX sample code.

## Choosing the Messages to Override

---

Which printing messages you need to override in your printer driver depends entirely on the characteristics of your device. Although QuickDraw GX provides default implementations of most messages (which means that you are not required to override them), there are some messages that you must override for raster or vector printing devices. Some drivers override many messages to provide their operations, and other drivers are created by overriding only a few messages.

### Note

The action of the default implementation of each printing message is noted in the chapter “Printing Messages.” Some of the default implementations provided by QuickDraw GX are empty (all that the function does is return). ♦

Your message overrides for a different printing device are likely to be quite different than those for the ImageWriter II, which are shown in Table 3-2 on page 3-13. For example, the driver for the LaserWriter IIg creates a PostScript file during the printing process. This driver only needs to override a few messages to perform its tasks, as shown in Table 3-3.

**Table 3-3** Message overrides for the LaserWriter IIg printer driver

---

Message	Why you override
GXPostScriptDoPageSetup	To add a PostScript setup string at the start of data for each page
GXOpenConnection	To create a file for the PostScript data
GXCloseConnection	To close the PostScript data file
GXInitialize	To allocate the message globals
GXDumpBuffer	To write the printer data to the file

The messages that your printer driver needs to override depend on the unique characteristics of your printing device. There is no set formula: what you have to do is familiarize yourself with the messages that are available from the printing system and look at the code for sample printer drivers. You can review the code for the sample drivers in the QuickDraw GX sample code, and you can examine the printing messages in the chapter “Printing Messages” in this book.

## Forwarding Messages

---

Your printer driver can forward a message in its implementation of a partial override. If you are totally overriding a message, you do not forward it to other message handlers. You can forward the message either before or after performing your own actions. To forward a message to the next message handler in the message chain, use a statement with the following format:



## Printer Drivers

```
anErr = Forward_MessageName(arguments);
```

For example, the ImageWriter II printer driver overrides the GXRasterLineFeed message to test whether it is printing in low resolution. If so, the override function, SD\_LineFeed, temporarily alters the value of one of the parameters, forwards the message, and then resets the parameter value. Listing 3-3 shows the override of the GXRasterLineFeed message from the ImageWriter II printer driver.

**Listing 3-3**      Overriding the GXRasterLineFeed message

```
OSErr SD_LineFeed (Int16 *lineFeedSize, Ptr buffer,
                  UInt32 *bufferPos,
                  gxRasterImageDataHdl hImageData )
{
    OSErr    anErr;
    Boolean  amLowRes;
    short    actualLineFeed = *lineFeedSize;

    amLowRes = ((*hImageData).vImageRes == ff(72));
/*
    If the user is printing in low-resolution mode, double the
    line-feed size because the ImageWriter line-feed commands are
    all expressed at 144 dpi.
*/
    if (amLowRes)
        *lineFeedSize <= 1;
/*
    To get rid of the "paper dance" for blank color passes,
    optimize the small motions into groups.
*/
    {
        SpecGlobalsHdl    hGlobals= GetMessageHandlerInstanceContext();
        SpecGlobalsPtr    pGlobals= *hGlobals;

        if ( (pGlobals->packagingOptions == kDoSmallLineFeeds) ||
            (*lineFeedSize < -1) ||
            (*lineFeedSize > 1) )
        {
            *lineFeedSize += pGlobals->lineFeeds;
            pGlobals->lineFeeds = 0;
            /* do the line feed in the default way */
            anErr = Forward_GXRasterLineFeed(lineFeedSize, buffer,
                                             bufferPos, hImageData);
        }
    }
}
```

## Printer Drivers

```

else
{
    pGlobals->lineFeeds += *lineFeedSize;
    *lineFeedSize = 0;
    anErr = noErr;
}

if (amLowRes)
    *lineFeedSize >= 1;
return(anErr);
}

```

The `SD_LineFeed` override function tests the printing resolution. If the user is printing at low resolution (72 dots per inch), then the line-feed value is temporarily doubled to accommodate the fact that all line feeds are expressed in units of 144 dots per inch on the ImageWriter II. This function then forwards the `GXRasterLineFeed` message to allow the default implementation to send the appropriate character sequences. The `GXRasterLineFeed` message is described on page 4-98 in the chapter “Printing Messages.”

## Sending Messages

---

Your printer driver can also send a printing message to other message handlers. When you send a message, QuickDraw GX receives it and then sends it to the first message handler in the chain. To send a message, use a statement with this format:

```
anErr = Send_GXMessageName(arguments);
```

For example, to send the `GXFreeBuffer` message, use the following statement:

```
anErr = Send_GXFreeBuffer(bufferPtr);
```

## Handling Exceptions in Your Message Overrides

---

The code samples presented in this chapter make use of an exception-handling strategy that simplifies the job of testing for error conditions after each function call. This strategy uses three C macros to branch to error handlers in response to conditions. The error-handling macros are described in the section “Handling Exceptions in Your Message Overrides” beginning on page 2-14 in the chapter “Printing Extensions.”

## Using the Printing-Related Collections

---

Much of the data that defines how a document is processed for printing is stored in collection objects. QuickDraw GX supports three collections to store printing-related information:


## Printer Drivers

- The job collection contains information from the Print dialog box.
- The format collection contains information from the Page Setup and Custom Page Setup dialog boxes.
- The paper-type collection stores information specified by the creator of the paper-type object.

These collections are described in *Inside Macintosh: QuickDraw GX Printing*. Collections and collection items are described in the chapter “Collection Manager” in *Inside Macintosh: QuickDraw GX Environment and Utilities*.

Most of the collection items with which you need to work with to develop a printer driver are found in the job collection, which is illustrated in Figure 3-2. You access the information in this collection when your driver needs to modify the choices that are presented to the user and when your driver needs to access the current settings of various printing parameters.

**Figure 3-2** The job collection



Job collection	
Print-job information	
Collation information	
Copy information	
Page-range information	
Quality information	
File-destination information	
File-location information	
File-format information	
File-fonts information	
Paper-feed information	
Tray-feed information	
Manual-feed information	
Standard mapping information	
Special mapping information	
Tray-mapping information	
Print-panel information	
Format-panel information	

## Printer Drivers

Each of the job collection items shown in Figure 3-2 is described in *Inside Macintosh: QuickDraw GX Printing*.

You can access each item in this collection by specifying its tag ID. Using tags to access collection items is described in *Inside Macintosh: QuickDraw GX Environment and Utilities*. The tag ID that you use for each item is listed in Table 3-4.

**Table 3-4** Tag ID constants for items in the job collection

Constant	Value	Explanation
gxJobTag	'job'	Print-job information
gxCollationTag	'sort'	Collation information
gxCopiesTag	'copy'	Copy information
gxPageRangeTag	'rang'	Page-range information
gxQualityTag	'qual'	Quality information
gxFileDestinationTag	'dest'	File-destination information
gxFileLocationTag	'floc'	File-location information
gxFileFormatTag	'ffmt'	File-format information
gxFileFontsTag	'incf'	File-fonts information
gxPaperFeedTag	'feed'	Paper-feed information
gxTrayFeedTag	'tray'	Tray-feed information
gxManualFeedTag	'manf'	Manual-feed information
gxNormalMappingTag	'nmap'	Standard mapping information
gxSpecialMappingTag	'smap'	Special mapping information
gxTrayMappingTag	'tmap'	Tray-mapping information
gxPrintPanelTag	'ppan'	Print-panel information
gxFormatPanelTag	'fpan'	Format-panel information

Each of the items in the job collection has a specific data structure associated with it, and most have a number of constants defined for setting the values of fields in the structures. These data structures and constants are described in the chapter “Page Formatting and Dialog Box Customization” in *Inside Macintosh: QuickDraw GX Printing*.

**▲ WARNING**

The size of the items in the job collection is subject to change as QuickDraw GX evolves. For that reason, it is important for you to specify an expected size for each item in your calls to the Collection Manager, rather than specifying `nil`, which tells the Collection Manager to copy the entire object no matter what its size is. The size that you specify must match the size of the data structure into which the item is being copied. ▲

The ImageWriter II printer driver accesses several of the job collection items to determine how to proceed with printing. One example, the `JobIsBest` function, is shown in Listing 3-12 on page 3-38. This function accesses the quality information to determine how to send data to the printing device.

## The QuickDraw GX ImageWriter II Printer Driver Messages

This section describes the messages and functions that make up the QuickDraw GX ImageWriter II printer driver. This driver needs to accomplish the following tasks, several of which are specific to raster printing:

- Initialize the environment.
- Modify the default printer object so that the application can choose to format for black-and-white or color printing at different resolution settings.
- Establish highest-resolution text printing as the preferred job format mode for the ImageWriter II.
- Update the printing-device configuration information and store it with the desktop printer.
- Respond to queries about direct-mode printing.
- Set up the data that is needed for printing the document. For the ImageWriter II, this means setting up information, including
  - the draft character table if the document is printing in text-draft mode
  - the halftone data for printing graphics
  - unidirectional or bidirectional printing, depending on the selected resolution
  - the raster packaging data, depending on the printing resolution
  - print-quality information
  - color-printing information
- Render each page by creating a buffer for the data and then sending it to the printing device.
- For each page in the document being printed, check to see if the page needs to be manually fed by the user. If so, alert the user to feed the appropriate paper type, and manage the user's interaction with the printing alert box.
- Shut down the environment at the end of printing.

## Printer Drivers

The messages that the ImageWriter II printer driver overrides are shown in Table 3-2 on page 3-13. The remainder of this section describes these messages and, as an example, shows how the ImageWriter II driver overrides them.

## Initializing the ImageWriter II Environment

---

QuickDraw GX sends the `GXInitialize` message when an application creates a job object, just after the application has called the `GXNewJob` function so that your driver can initialize any storage and establish the values of global variables. The ImageWriter II printer driver performs very simple initialization. Its override of the `GXInitialize` message, `SD_Initialize`, allocates a globals world so that the driver can access the QuickDraw GX globals. The `SD_Initialize` function is shown in Listing 3-4.

---

**Listing 3-4**      Initializing the ImageWriter II printer driver

```
OSErr SD_Initialize (void)
{
    SpecGlobalsHdl hGlobals;
    OSErr          anErr;

    /* create the globals */
    hGlobals = (SpecGlobalsHdl) NewHandleClear(
                                                sizeof(SpecGlobals) );

    anErr = MemError();

    /* save the globals */
    SetMessageHandlerInstance(hGlobals);

    /* branch to exception handler if necessary */
    nrequire( anErr, MNewHandleClear );

    /* no draft table allocated yet */
    (**hGlobals).draftTable = nil;

    return(noErr);

MNewHandleClear:
    return(anErr);
}
```

The `SD_Initialize` function first allocates the QuickDraw GX message globals. Then it initializes the global handle to the draft table to `nil` because this table, which is used for drawing draft-quality characters on the printing device, is only needed when the ImageWriter II is printing in draft mode.

You almost always override the `GXInitialize` message for a driver. No matter which imaging system your driver is written for, you need to initialize the QuickDraw GX message globals. The `GXInitialize` message is described on page 4-43 in the chapter “Printing Messages.”

## Providing the Application With Printing Options

QuickDraw GX sends the `GXDefaultPrinter` message when an application creates a job object so that your driver can create new view devices that represent the specific features provided by your printing device. These features include color information. For raster printing devices, the features also include resolution variations; for vector printing devices, the features include pen information. You can override this message to create view devices for your driver. You attach these view devices to the job object, and the application can then access them to determine specific device settings.

The ImageWriter II printer driver provides various printing resolutions and allows the application to print either in black and white or in color. The driver overrides the `GXDefaultPrinter` message to add two high resolution (144 dpi) view devices for use by applications. This message override, the `SD_DefaultPrinter` function, is shown in Listing 3-5. The `GXDefaultPrinter` message is described on page 4-50 in the chapter “Printing Messages.”

**Listing 3-5**      Modifying the default printer object

```
OSErr SD_DefaultPrinter(gxPrinter thePrinter)
{
    OSErr      anErr;
    gxViewDevice vd;

    /* add the standard view devices first */
    anErr = Forward_GXDefaultPrinter(thePrinter);
    nrequire(anErr, DefaultPrinter);

    /* add a 144 b/w view device */
    vd = NewDeviceResolutionViewDevice();

    {
        gxSetColor theColors[2];
        gxSetColor *pColor;
        gxColorSet theSet;
```

## Printer Drivers

```

pColor = &theColors[0];

pColor->rgb.red    = 0xFFFF;
pColor->rgb.green  = 0xFFFF;
pColor->rgb.blue   = 0xFFFF;

pColor++;
pColor->rgb.red    = 0x0000;
pColor->rgb.green  = 0x0000;
pColor->rgb.blue   = 0x0000;

theSet = GXNewColorSet(gxRgbSpace, 2, theColors);
SetViewDeviceColorSet(vd, theSet);
GXDisposeColorSet(theSet);
}

anErr = GXAddPrinterViewDevice(thePrinter, vd);
nrequire(anErr, FailedAddBWViewDevice);

/* add a 144 color view device with 8 colors */
if (PrinterHasColorRibbon())
{
    gxSetColor  theColors[8];
    gxSetColor  *pColor;
    gxColorSet  theSet;
    short       idx;

    vd = NewDeviceResolutionViewDevice();
    pColor = &theColors[0];
    for (idx = 0; idx < 8; ++idx)
    {
        /* default the color to black */
        pColor->rgb.red    = 0x0000;
        pColor->rgb.green  = 0x0000;
        pColor->rgb.blue   = 0x0000;
        /* then fill in the RGB components */
        if (idx & 0x04)
            pColor->rgb.red    = 0xFFFF;
        if (idx & 0x02)
            pColor->rgb.green  = 0xFFFF;
        if (idx & 0x01)
            pColor->rgb.blue   = 0xFFFF;
        ++pColor;
    }
}

```



Printer Drivers

```
        /* move onto the next color */
    }
    theSet = GXNewColorSet(gxRgbSpace, 8, theColors);
    SetViewDeviceColorSet(vd, theSet);
    GXDisposeColorSet(theSet);

    anErr = GXAddPrinterViewDevice(thePrinter, vd);
    nrequire(anErr, FailedAddColorViewDevice);
}

return(noErr);

/* exception handling */
FailedAddColorViewDevice:
FailedAddBWViewDevice:
    GXDisposeViewDevice(vd);

GXDefaultPrinter:
    return(anErr);
}
```

The `SD_DefaultPrinter` function first forwards the `GXDefaultPrinter` message so that the standard view devices can be added, then adds two of its own: a 144 dpi 1-bit view device and a 144 dpi 8-color view device. This function uses a `for` loop to assign color values that correspond to the eight basic RGB colors. These values are shown in Table 3-5.

**Table 3-5** Color values for an eight-color view device for the ImageWriter II printer

Color name	Color index	Red value	Green value	Blue value
White	0	0xFFFF	0xFFFF	0xFFFF
Yellow	1	0xFFFF	0xFFFF	0x0000
Magenta	2	0xFFFF	0x0000	0xFFFF
Red	3	0xFFFF	0x0000	0x0000
Cyan	4	0x0000	0xFFFF	0xFFFF
Green	5	0x0000	0xFFFF	0x0000
Blue	6	0x0000	0x0000	0xFFFF
Black	7	0x0000	0x0000	0x0000

The `SD_DefaultPrinter` function calls two local functions, the code for which is found in the QuickDraw GX sample code:

## Printer Drivers

- The `NewDeviceResolutionViewDevice` function creates a view device object and sets up its mapping for 144 dpi.
- The `PrinterHasColorRibbon` function returns `true` if the desktop printer information says that the printing device has a color ribbon installed on it.

## Establishing the Preferred Printing Characteristics

---

QuickDraw GX sends the `GXDefaultJob` message to initialize a new job object. You can override this message, which is described on page 4-47 in the chapter “Printing Messages,” to add collections to the job or to establish printing preferences for the job.

The ImageWriter II printer driver overrides the `GXDefaultJob` message to add an item to the job collection that stores the preferred printing resolution and to initialize that preference to the highest resolution possible. This message override, the `SD_DefaultJob` function, is shown in Listing 3-6.

---

**Listing 3-6**      Establishing the printing resolution for the ImageWriter II printer

```
OSErr SD_DefaultJob()
{
    OSErr anErr;

    anErr = Forward_GXDefaultJob();
    if (anErr == noErr)
    {
        long imagewriterOptions = kSuperRes;
        /* add high resolution preference item */
        anErr = AddCollectionItem(GXGetJobCollection(GXGetJob()),
                                DriverCreator, 0, sizeof(imagewriterOptions),
                                &imagewriterOptions);
    }
    return(anErr);
}
```

The `SD_DefaultJob` function adds an item to the job collection. Other printer drivers override the `GXDefaultJob` message for similar purposes. For example, one vector printer driver overrides this message to create a number of collection items that are used in its other functions. These items, which are added to the job collection, include the current carousel list, pen list, pens dialog box settings, and plot-quality settings.

When a user chooses the Page Setup dialog box, QuickDraw GX sends the `GXJobDefaultFormatDialog` message, which you can override to modify the contents of the dialog box. The ImageWriter II printer driver override of this message, `SD_JobFormatDialog`, determines if the job includes support for using the text-mode printing capabilities of the ImageWriter II, as shown in Listing 3-7. If the job does support text-mode printing, then text mode becomes the preferred printing mode.

The `GXJobDefaultFormatDialog` message is described on page 4-82 in the chapter “Printing Messages.”

---

**Listing 3-7** Determining the preferred job-formatting mode

```

OSError SD_JobFormatDialog(gxDialogResult *theResult)
{
    OSError          anErr;
    gxJobFormatModeTableHdl theJobFormatModeList;
    long             i;
    gxJob            theJob = GXGetJob();

    /* set up the job format mode information */

    anErr = GXGetAvailableJobFormatModes(&theJobFormatModeList);
    if (!anErr) && (theJobFormatModeList)
    {
        for (i = 0; i <= (*theJobFormatModeList)->numModes - 1; ++i)
        {
            if ((*theJobFormatModeList)->modes[i] ==
                gxTextJobFormatMode)
            {
                GXSetPreferredJobFormatMode(gxTextJobFormatMode,
                                             false);

                break;
            }
        }
        DisposHandle((Handle)theJobFormatModeList);
    }

    /*do normal dialogs after handling job format mode stuff */
    return(Forward_GXJobDefaultFormatDialog(theResult));
}

```

The `SD_JobFormatDialog` function calls the `GXGetAvailableJobFormatModes` function to determine which formatting modes the application supports. If text formatting mode is supported, `SD_JobFormatDialog` makes that the preferred 0mode by calling the `GXSetPreferredJobFormatMode` function. The `GXGetAvailableJobFormatModes` function is described on page 5-30 and the `GXSetPreferredJobFormatMode` function is described on page 5-30 in the chapter “Printing Functions for Message Overrides.”

## Storing the Current Printer Configuration

---

QuickDraw GX sends the `GXOpenConnection` message to open a connection with a printing device. You can override this message, which is described on page 4-131 in the chapter “Printing Messages,” to perform any special operations that you need to do at the time of connection.

The ImageWriter II printer driver overrides the `GXOpenConnection` message to update the printer configuration that is stored with the desktop printer. It first sends a query to the printing device for its hardware configuration, which includes information about the sheet-feeder and color-ribbon options. It then stores that information with the desktop printer. This message override, the `SD_OpenConnection` function, is shown in Listing 3-8.

---

**Listing 3-8**      Opening the connection with the printing device

```
OSErr SD_OpenConnection(void)
{
    OSErr    anErr;

    /* first, open the connection the standard way */
    anErr = Forward_GXOpenConnection();
    nrequire(anErr, OpenConnection);

    /*
       then, bring the desktop-printer configuration information
       up to date
    */
    anErr = UpdateConfiguration();
    nrequire(anErr, UpdateConfiguration);

    return(noErr);

    /* exception handling */
UpdateConfiguration:
    GXCleanupOpenConnection();
OpenConnection:

    return(anErr);
}
```

The `SD_OpenConnection` function queries and stores the hardware configuration by calling a local function, `UpdateConfiguration`. This function, which is shown in Listing 3-9, calls the `FetchStatusString` function to query the ImageWriter II and then stores the information in the printer configuration file.

**Listing 3-9** Getting information about the configuration of the printing device

```

OSErr UpdateConfiguration(void)
{
    Str32                deviceName;
    OSErr                anErr = noErr;
    ImageWriterConfigHandle configHandle;
    ImageWriterConfigPtr  configPtr;
    Boolean              isImageWriterII = false;
    ResType               commType;

    /* find out printer name and how the printer is connected */
    GXGetPrinterName(GXGetJobOutputPrinter(GXGetJob()),
                    deviceName);
    anErr = GXFetchDTPData(deviceName, gxDeviceCommunicationsType,
                          gxDeviceCommunicationsID, (Handle*)&configHandle);
    nrequire(anErr, FetchCommType);
    commType = **(ResType**)configHandle;
    DisposHandle((Handle) configHandle);

    /* store the communications type for future use */
    {
        SpecGlobalsHdl hGlobals = GetMessageHandlerInstanceContext();

        (**hGlobals).commType = commType;
    }

    /* find out the original configuration */
    if (GXFetchDTPData(deviceName, kImageWriterConfigType,
                      kImageWriterConfigID, (Handle*)&configHandle) == noErr)
    {
        /* remember that it was an IW2 */
        configPtr = *configHandle;
        isImageWriterII = configPtr->isImageWriterII;
        DisposeHandle((Handle) configHandle);

        /*
         * If this is not an ImageWriter II, bail out now
         * because the timeout takes two minutes!
         */
        if (!isImageWriterII)
            return(noErr);
    }
}

```

## Printer Drivers

```

else
{

    /* if not sure yet, assume IW2 for PAP and IW1 otherwise */
    if (commType == 'PPTL')
        isImageWriterII = true;
}

/* make a handle to hold configuration info for the printer */
configHandle = (ImageWriterConfigHandle)
    NewHandle(sizeof(ImageWriterConfigRecord) );
anErr = MemError();
nrequire(anErr, NewHandle);

/* set up the default for the device */
configPtr = *configHandle;
configPtr->hasColorRibbon = false;
configPtr->hasSheetFeeder = false;
configPtr->isImageWriterII = true;

{
short    statusReturn;

/* query the device */
anErr = FetchStatusString(&statusReturn, (commType == 'PPTL'));

/*
    Scan the status string looking for information about
    printer kind and options.
*/
configPtr = *configHandle;
if ( anErr == gxAioTimeout )
{
    /*
        If you don't know what kind of printer it is and the
        request timed out, assume that it is an IW1.
    */
    if (!isImageWriterII)
    {
        anErr = noErr;
        configPtr->isImageWriterII = false;
    }
}
}

```

## Printer Drivers

```

else
{
    configPtr->hasColorRibbon =
        (statusReturn & (0x1000 >> kColorRibbonBit)) != 0;
    configPtr->hasSheetFeeder =
        (statusReturn & (0x1000 >> kSheetFeederBit)) != 0;
}
nrequire(anErr, FetchStatusString);
}

/* write out the new configuration */
anErr = GXWriteDTPData(deviceName, kImageWriterConfigType,
                        kImageWriterConfigID, (Handle)configHandle);

/* exception handling */
FetchStatusString:
    DisposHandle((Handle) configHandle);

NewHandle:
FetchCommType:
    return(anErr);
}

```

The `UpdateConfiguration` function first calls the `GXFetchDTPData` function to access the communications type that is stored for the printer with the desktop printer. `UpdateConfiguration` next calls `GXFetchDTPData` to determine with which kind of printer it is communicating. Then, `UpdateConfiguration` calls the `FetchStatusString` function. This is a local function that sends a query to the hardware device and receives a status string back from it. `UpdateConfiguration` examines the status string to determine whether the printer has a sheet feeder or a color ribbon. Finally, `UpdateConfiguration` calls the `GXWriteDTPData` function to store the information that it has updated in the desktop printer.

Desktop printers are described in *Inside Macintosh: QuickDraw GX Printing*. The `GXFetchDTPData` function is described on page 5-27 and the `GXWriteDTPData` function is described on page 5-26 in the chapter “Printing Functions for Message Overrides.”

## Responding to Direct-Mode Queries

QuickDraw GX sends the `GXJobFormatModeQuery` message to get or set format-mode information. This message only applies to direct-mode printing, which the user can select for printing to a specific printing device such as the ImageWriter II. Direct mode

## Printer Drivers

allows for accelerated printing of text for printing devices such as the ImageWriter II; however, illustrations cannot be printed while text direct-mode printing is selected.

QuickDraw GX can make various requests through the `GXJobFormatModeQuery` message that you need to respond to, including requests

- to get a list of the fonts that are supported for the job
- to get a list of the styles that are supported for the job
- to get the positioning constraints of the job for font sizes and for line drawing
- to set the current font style, which can be chosen from the list of supported styles

These requests apply only to text and line printing mode. The `GXJobFormatModeQuery` message is described on page 4-59 in the chapter “Printing Messages.” For a complete list and description of the `GXJobFormatModeQuery` request types, see the chapter “Advanced Printing Features” in *Inside Macintosh: QuickDraw GX Printing*.

The ImageWriter II printer driver overrides the `GXJobFormatModeQuery` message to implement its responses to the queries. Its implementation is the `SD_JobFormatModeQuery` function, a portion of which is shown in Listing 3-10.

---

**Listing 3-10**    Responding to a query about the job format mode

```
OSErr SD_JobFormatModeQuery( gxQueryType theQuery,
                             void* srcData, void* dstData)
{
    OSErr    anErr = noErr;
    Handle    theFonts;
    Handle    theStyles;

    switch(theQuery)
    {
        case gxSetStyleJobFormatCommonStyleQuery:
        {
            char        *pStyleName;
            /* fetch the list of supported styles */
            anErr = Send_GXFetchTaggedDriverData('STR#',
                                                kFormatModeStylesID, &theStyles);
            require(anErr == noErr, FailedToLoadStyles1);
            HNoPurge(theStyles);
            HLock(theStyles);

            /*
                Determine which style is being referenced and set
            */
        }
    }
}
```



## Printer Drivers

```

        the corresponding style (only two styles are
        currently supported).
    */

    if (**((short **) theStyles) == 2)
    { /* the correct number of styles are in place */
        char whichFace = 0;

        pStyleName = ((char *) *theStyles) + sizeof(short);

        if ( IUCompString(pStyleName, (char *) srcData) == 0 )
        { /* user wants bold face */
            whichFace = bold;
        }
        else
        { /* point to next name in the list */
            pStyleName += *pStyleName + 1;
            if ( IUCompString(pStyleName,
                             (char *) srcData) == 0 )
            { /* user wants the underline face */
                whichFace = underline;
            }
        }
    }

    /* if user specified a valid face, set it now */
    if (whichFace != 0)
    {
        SetStyleCommonFace((gxStyle) dstData,
                           GetStyleCommonFace((gxStyle) dstData) | whichFace);
    }
}
/* else - something is wrong with our resource */

DisposHandle(theStyles); /* dump temporary handle */

break;
}

```

The `SD_JobFormatModeQuery` function is designed as a switch statement, with code similar to that of Listing 3-10 (for the `gxSetStyleJobFormatCommonStyleQuery` request) included for each of the queries. The full version of this function is found in the QuickDraw GX sample code.

## Setting Up the Parameters for Printing

---

QuickDraw GX sends the `GXSetupImageData` message so that you can initialize any constant data that your driver needs to use for imaging a document. The ImageWriter II driver overrides this message with the `SD_SetupImageData` function, which sets up the following data for a print job:

- the draft character table for printing in text mode
- halftone data for printing in graphics mode
- the start-of-page string for unidirectional or bidirectional printing, depending on the print quality of the print job
- the raster packaging information, which is different for different qualities of printing
- print-quality information
- color-printing information

The `GXSetupImageData` message is described on page 4-92 in the chapter “Printing Messages.” A portion of the `SD_SetupImageData` function is shown in Listing 3-11. The complete version of this function is found in the QuickDraw GX sample code.

---

**Listing 3-11**     Setting up the constant data for the print job

```
OSErr SD_SetupImageData( gxRasterImageDataHdl hImageData )
{

    OSErr          anErr;
    gxRasterImageDataPtr pImageData;
    Boolean         isJobNotFinalQuality, isTextJobFormatMode;
    long            imagewriterOptions;

    /* do the default setup */
    anErr = Forward_GXSetupImageData(hImageData);
    nrequire(anErr, Forward_GXSetupImageData);

    /* test for 'final' quality mode */
    isJobNotFinalQuality = !JobIsBest(&imagewriterOptions);

    /* test for gxTextJobFormatMode */
    isTextJobFormatMode = ( GXGetJobFormatMode( GXGetJob() ) ==
                           gxTextJobFormatMode );

    /*
       If the job is not final quality and is not using text mode,
       downgrade the imaging data to lower quality.
    */
}
```

## Printer Drivers

```

if (isJobNotFinalQuality || isTextJobFormatMode)
{
    /* rough or text mode */
    pImageData = *hImageData; /* dereference for size+speed */

    /* image at 80 or 72 dpi */
    if (imagewriterOptions & kSuperRes)
        pImageData->hImageRes = ff(80);
    else
        pImageData->hImageRes = ff(72);
    pImageData->vImageRes = ff(72);

    /*
     * If using text mode, load the draft table;
     * otherwise, set up the halftone data.
     */
    if (isTextJobFormatMode)
    { /* load the draft table */
        Handle draftTable;
        SpecGlobalsHdl hGlobals =
            GetMessageHandlerInstanceContext();

        anErr = Send_GXFetchTaggedDriverData('idft',
            gxPrintingDriverBaseID, &draftTable);
        nrequire(anErr, FailedToLoadDraftTable);

        (**hGlobals).draftTable = draftTable;
    }
    else
    {
        /*
         * Usedither level that looks better at 72 dpi
         * than do the default values from the resources.
         */
        pImageData->theSetup.planeSetup[0].planeHalftone.method
            = 4;

        /* turn off color matching when in non-final mode */
        pImageData->theSetup.planeSetup[0].planeProfile = nil;
    }
}
...

```

The remainder of the `SD_SetupImageData` function operates similarly, modifying the default values for varying resolution values.

## Printer Drivers

The `JobIsBest` function that is called by `SD_SetupImageData` accesses the job collection to establish the print quality. The code for this function is shown in Listing 3-12. The quality information is stored as an item in the job collection that specifies how the user wants the job printed. This information is accessed from the job collection using the `gxQualityTag` tag ID.

---

**Listing 3-12** Establishing the print quality

```
Boolean JobIsBest(long *imagewriterOptions)
{
    Boolean          isFinal;
    gxQualityInfo    jobQualitySettings;
    long             itemSize = sizeof(jobQualitySettings);
    OSErr            status;
    Collection        jobCollection;

    /* cache the collection */
    jobCollection = GXGetJobCollection(GXGetJob());

    isFinal = false;
    status = GetCollectionItem(jobCollection, gxQualityTag,
                              gxPrintingTagID, &itemSize, &jobQualitySettings);

    if ( (status == noErr) && (jobQualitySettings.currentQuality
                              == (jobQualitySettings.qualityCount-1)) )
        isFinal = true;

    /* default is kSuperRes */
    *imagewriterOptions = kSuperRes;
    itemSize = sizeof(imagewriterOptions);
    status = GetCollectionItem(jobCollection, DriverCreator, 0,
                              &itemSize, imagewriterOptions);

    return(isFinal);
}
```

The `JobIsBest` function returns `true` if the current print job is printed in final quality; otherwise, it returns `false`. It also accesses and returns the ImageWriter II options from the job collection.

## Managing Special Page Handling

QuickDraw GX sends the `GXStartSendPage` message during imaging, just before each page is rendered. The ImageWriter II printer driver overrides this message with the `SD_StartSendPage` function to manage manual feeding of paper. The `SD_StartSendPage` function first checks to see if the entire print job uses automatic paper feeding; if not, it determines if the paper type of the page that is about to be rendered is a manually fed paper type. If so, `SD_StartSendPage` displays a printing alert box, as described in the section “Displaying Status Information and the Printing Alert Boxes” beginning on page 3-41.

The first portion of the `SD_StartSendPage` function uses the job collection to determine if the entire print job uses automatic paper feeding, as shown in Listing 3-13. The `GXStartSendPage` message is described on page 4-136 in the chapter “Printing Messages.”

**Listing 3-13** Determining if the print job uses any manually fed pages

```
{
    OSErr          anErr = noErr;
    gxJob          theJob = GXGetJob();
    Collection      jobCollection;
    gxPaperFeedInfo paperFeed;
    long           itemSize = sizeof(paperFeed);
    ResType         commType;
    short           statusReturn;

    jobCollection = GetJobCollection(theJob);

    /* cache the communications type */
    commType = (**(SpecGlobalsHdl)
                GetMessageHandlerInstanceContext()).commType;
    if (commType == 'PPTL')
    {
        FetchStatusString(&statusReturn, true, true);
        nrequire(anErr, FetchStatusString);
    }
    else
        statusReturn = 0;

    /* first, determine if the entire job is auto feed */
    paperFeed.autoFeed = true;
    (void) GetCollectionItem(jobCollection, gxPaperFeedTag,
                            gxPrintingTagID, &itemSize, &paperFeed);
    /*
```

## Printer Drivers

If the entire job is not auto feed, the value of the `paperFeed.autoFeed` field will now be changed to false.

```
*/
.....
```

If the `GetCollectionItem` call succeeds in finding a paper-feed item, then the value of the `paperFeed.autoFeed` field is determined by the retrieved value. Otherwise, the value of the `paperFeed.autoFeed` field remains true, as it was before making the call. If this field is false, the job includes manual feeding.

The next portion of the `SD_StartSendPage` function executes if the print job includes any manually fed pages. This portion loops through the list of manual-feed paper-type names and uses the job collection to determine if the entire job uses automatic paper feeding, as shown in Listing 3-14.

---

**Listing 3-14** Finding the manual-feed paper name

```
if (!paperFeed.autoFeed)
{
    /* get the manual-feed list, using the paper-feed size */
    gxManualFeedInfo **feedHandle;

    feedHandle = (gxManualFeedInfo**) NewHandle(0);
    anErr = MemError();
    nrequire(anErr, FailedNewHandle);

    anErr = GetCollectionItemHdl(jobCollection, gxManualFeedTag,
                                gxPrintingTagID, (Handle)feedHandle);
    if (anErr == noErr)
    {
        Str31          paperName;
        short          idx;
        gxManualFeedInfo *pFeed;

        /* get name of this page's paper type */
        GXGetPaperTypeName(GXGetFormatPaperType(pageFormat),
                           paperName);

        paperFeed.autoFeed = true;

        /* lock and dereference for the loop */
        HLockHi((Handle) feedHandle);
        pFeed = *feedHandle;
        /* look for the manually fed paper-type name */
    }
}
```

```

        for (idx = 0; idx < pFeed->numPaperTypeNames; ++idx)
        {
            Ptr pName = &pFeed->paperTypeNames[idx];

            if (IUMagIDString( paperName, pName,
                               paperName[0], *pName+1) == 0)
            {
                paperFeed.autoFeed = false;
                break;
            }
        }

        DisposHandle((Handle) feedHandle);
        FailedNewHandle:
        ;
    }

```

If the manual-feed paper-type name is found (if `paperFeed.autoFeed` is false after exiting the loop) or if the paper-name list is empty, then the remainder of `SD_StartSendPage` displays a printing alert box to tell the user to manually feed the appropriate paper. This portion of the `SD_StartSendPage` function is shown in the next section. The code for the entire `SD_StartSendPage` function is found in the QuickDraw GX sample code.

## Displaying Status Information and the Printing Alert Boxes

You can report status information to the user during the printing process to keep the user informed. Informative text about the printing status is displayed in the desktop printer window.

In addition, when the user has to perform an action before printing can continue, your driver may need to display a printing alert box. You process a printing alert following this sequence of actions:

1. Make a status record that contains the request to the user.
2. Call the `GXAlertTheUser` function to display the status information in a printing alert box. The `GXAlertTheUser` function is described on page 5-18 in the chapter “Printing Functions for Message Overrides.”
3. Keep sending the printing alert (repeatedly call `GXAlertTheUser`) until one of three actions occurs:
  - ☐ The condition resolves itself.
  - ☐ The user responds by clicking a button in the printing alert box.
  - ☐ An error of some sort occurs.

## Printer Drivers

4. If necessary, call `GXAlertTheUser` again with other informational text so that the user can dismiss the printing alert box. You need to do this if an error occurred or if the problem resolved itself. If the user dismisses the printing alert box, you don't need to take this step.

In the ImageWriter II printer driver, the `SD_StartSendPage` function displays a printing alert box when the user has to manually feed a page into the printing device, as shown in Listing 3-15.

**Listing 3-15**     Displaying a printing alert box with printer status information

```
{
    StatusRecordPtr    pStat;

    /* make a status record with the request to the user */
    pStat = (StatusRecordPtr) NewPtrClear(sizeof(gxStatusRecord) +
                                         sizeof(gxManualFeedRecord));

    anErr = MemError();
    nrequire(anErr, NewPtrClear);
    /* use the built-in status resource for this */
    pStat->statResId = gxUnivAlertStatusResourceId;
    pStat->dialogResult = nil;

    if (!paperFeed.AutoFeed)
    {
        gxManualFeedRecord *pFeed;

        /* use the provided manual-feed alert text string */
        pStat->statResIndex = gxUnivManualFeedIndex;
        pStat->bufferLen = sizeof(gxManualFeedRecord);
        pFeed = (gxManualFeedRecord*)&pStat->statusBuffer;
        /* the user can choose to switch to auto feed */
        pFeed->canAutoFeed = true;
        GXGetPaperTypeName(GXGetFormatPaperType(pageFormat),
                           pFeed->paperTypeName);
    }
    else
    {
        gxOutOfPaperRecord *pOut;

        pStat->statResIndex = gxUnivOutOfPaperIndex;
        pStat->bufferLen = sizeof(gxOutOfPaperRecord);
        pOut = (gxOutOfPaperRecord *)&pStat->statusBuffer;
        GXGetPaperTypeName(GXGetFormatPaperType(pageFormat,
```



## Printer Drivers

```

                                pOut->paperTypeName);
    }

do /* loop, sending the alert until it gets resolved */
{
    anErr = GXAlertTheUser(pStat);

    /* if the paper suddenly got loaded, do an OK */
    if (commType == 'PPTL')
    {
        FetchStatusString(&statusReturn, true);
        if ((statusReturn & kOutOfPaperMask) == 0)
        {
            pStat->dialogResult = ok;
            anErr = noErr;
        }
    }
} while ((anErr == noErr) && (pStat->dialogResult == nil));

/* decide what to do, based on the user's response */
switch ( pStat->dialogResult )
{
    case ok:                    /* the paper is now loaded */
        break;

    case cancel:                /* user wants to stop printing */
        anErr = gxPrUserAbortErr;
        break;

    case gxAutoFeedButtonId: /* do rest of job with auto feed */
        paperFeed.gxAutoFeed = true;
        (void) AddCollectionItem(jobCollection, gxPaperFeedTag,
                                gxPrintingTagID, itemSize, &paperFeed);
        break;
}

DisposPtr((Ptr) pStat); /* done with status now, so dispose */
}

/* now display the "Sending data to the printer" text */
if (anErr == noErr)
    anErr = GXReportStatus(kDriverStatus, kSendingData);
...
}

```

## Printer Drivers

This printing alert box makes use of one of the built-in alert conditions that are defined for printer drivers. Table 3-6 lists the predefined alert conditions, each of which is an index into the corresponding informational text strings stored in a predefined printing alert ('plrt') resource.

**Table 3-6** Predefined alert conditions for printing device drivers

Constant	Value	Explanation
<code>gxUnivManualFeedIndex</code>	2	Paper needs to be manually fed into the printing device
<code>gxUnivFailToPrintIndex</code>	3	The printing device could not print the job
<code>gxUnivPaperJamIndex</code>	4	A paper jam has occurred on the printing device
<code>gxUnivOutOfPaperIndex</code>	5	The printing device is out of paper
<code>gxUnivNoPaperTrayIndex</code>	6	The required paper tray is not in place
<code>gxUnivPrinterReadyIndex</code>	7	The printing device is ready to print

You can add your own informational text to use with the `GXAlertTheUser` function by defining printing alert ('plrt') resources, which are described in the chapter "Printing Resources" in this book.

### Checking for When an Alert Condition Resolves Itself

Listing 3-15 on page 3-42 includes a loop that continues to call `GXAlertTheUser` to display the printing alert box until the user feeds the paper. If you are using a printing device that can automatically detect paper feeding, you could change the loop as shown in Listing 3-16.

**Listing 3-16** Checking if an alert condition has resolved itself

```
do
{
    anErr = GXAlertTheUser(pStatus);
    /* see if the user has loaded the paper */
    if ((anErr == noErr) && (pStatus->dialogResult == nil))
    {
        Boolean userFedPaper;

        userFedPaper = CheckToSeeIfPaperWasFed();
        if (userFedPaper)
        {
```

```

        pStatus->statResIndex = gxUnivPrinterReadyIndex;
        anErr = GXAlertTheUser(pStatus);
        pStatus->dialogResult = ok;
    }
}
} while ((pStatus->dialogResult == nil) && (anErr == noErr));

```

In this version of the alert loop, the driver calls the `CheckToSeeIfPaperWasFed` function, which detects if the user has fed a page of paper into the printing device. When this happens, the manual-feed text string in the printing alert box is replaced with the string “Printer is ready,” which the user can hide at any time.

### Filling In Alert Information at Run Time

When you display a printing alert box and you need to dynamically fill in parts of the alert text string, you have to override two messages:

- Override the `GXInitializeStatusAlert` message to fill in the printing alert box at run time. This message is described on page 4-164 in the chapter “Printing Messages.”
- Override the `GXHandleAlertEvent` message to manage what happens when an event occurs in the printing alert box. This message is described on page 4-165 in the chapter “Printing Messages.”

You can also override the `GXHandleAlertFilter` message to work with any controls that have been installed in the printing alert box. This message is described on page 4-168 in the chapter “Printing Messages.”

Listing 3-17 shows portions of the overrides for these messages that could be used to tell the user that a new pen carousel needs to be placed in a plotter.

**Listing 3-17** Modifying alert information at run time

```

OSErr MyInitializeStatusAlert( StatusRecordPtr pStatus,
                             DialogPtr *pDialog )
{
    OSErr anErr = noErr;

    /* first see if this message is for your driver */
    if (pStatus->statusOwner == kDrvrCreatorType)
    {
        if (pStatus->statusId == kChangeCarouselsDlogID)
        {
            *pDialog = GetNewDialog( kChangeCarouselsDlogID, nil,
                                    (WindowPtr)-1);

            if (*pDialog == nil)
                anErr = resNotFound;
        }
    }
}

```

## Printer Drivers

```

        else      /* fill in run-time information */
            FillInDialogStrings( pStatus, pDialog );
    }
    else...      /* handle other status conditions */

    }
    else          /* the status text belongs to someone else */
        Forward_GXInitializeStatusAlert( pStatus, pDialog );

    return( anErr );
}

OSErr MyHandleAlertEvent( StatusRecordPtr pStatus, DialogPtr
                        *pDialog, EventRecord *theEvent, short *itemHit )
{
    OSErr anErr = noErr;

    /* first see if this message is for your driver */
    if (pStatus->statusOwner == kDrvrCreatorType)
    {
        if (pStatus->statusId == kChangeCarouselsDlogID)
        {
            HandleCarouselDialogEvent( pDialog, theEvent, itemHit );
            pStatus->dialogResult = *itemHit;
        }
        else.../* handle other status condition events */

    }
    else      /* the status message belongs to someone else */
        Forward_GXHandleAlertEvent( pStatus, pDialog, theEvent,
                                    itemHit );

    return( anErr );
}

```

Each of these functions first checks if the status condition is one that it handles. If not, the condition is passed on to the next handler in the message chain. If the condition is one that the driver handles, the function performs the needed operation: either filling in the name of the carousel that needs to be displayed in the printing alert box or handling an event in the alert box. If the driver handles the alert in these functions, it does not forward the message to the other message handlers.

## Displaying Status Text During Printing

When you want to display status information to the user and it does not require a response, you can call the `GXReportStatus` function, providing it with the ID of a status resource and the ID of the static text in that resource that you want sent to the desktop printer window.

QuickDraw GX provides several status ('stat') resources with a number of predefined text strings that you can display to the user. The printer-status text strings, which are listed in Table 3-7, are located in the page transmission status resource. Some of these strings are repeated, with one entry associated with an alert condition, and the other used to display informational status.

**Table 3-7** Status text IDs in the page transmission status resource

Text ID	Text string
1	“Waiting for the next sheet of paper...”
2	“Sending part of the page...”
3	“Preparing part of the page...”
4	“The printer is out of paper.”
5	“The printer has a paper jam.”
6	“The paper tray is improperly loaded.”
7	“The printer door is open. Please close it.” (alert version)
8	“The printer door is open. Please close it.” (status version)
9	“A print test is in progress. Please wait...”
10	“The printer fixing unit is being heated. Please wait...”
11	“The toner cartridge is improperly loaded.” (alert version)
12	“The toner cartridge is improperly loaded.” (status version)
13	“Printing the page.”
14	“Trying to locate the printer...”
15	“Opening a connection to the printer...”

## Rendering the Page on the Printing Device

QuickDraw GX sends the `GXRenderPage` message once for each page, when the QuickDraw GX shape that is the picture of the page needs to be translated into a stream of data that the printing device can use to render the output page. QuickDraw GX's default implementation for this function handles most of the work for you. For a raster printing device, it converts the page into bitmap data and sends the `GXRasterPackageBitmap` message before sending the bitmap to the printing device.

## Printer Drivers

When the raster printing-device head has to be moved down the page, QuickDraw GX sends the `GXRasterLineFeed` message.

Printing is accomplished by adding bytes of data to a buffer that you send to the printing device with the `GXBufferData` message.

The ImageWriter II printer driver overrides the `GXRenderPage` message to add its own operations. Its override function, `SD_RenderPage`, determines if the page is to be printed in text mode or not. If so, it calls the local function `PrintPageInDraftMode` to do the printing. If the page is printed in graphics mode, `SD_RenderPage` sends escape sequences to the printing device to set up the page size and then forwards the `GXRenderPage` message so that the default implementation can manage the page printing. The `GXRenderPage` message is described on page 4-96 in the chapter “Printing Messages.”

The ImageWriter II printer driver overrides the `GXRasterLineFeed` message with the `SD_LineFeed` function. This function adjusts the line-feed size if the driver is printing at low resolution because the ImageWriter II assumes that line-feed commands are expressed in high-resolution values. `SD_LineFeed` forwards the `GXRasterLineFeed` message to allow the default implementation to send the appropriate escape sequences to the printing device. The `GXRasterLineFeed` message is described on page 4-98 in the chapter “Printing Messages.”

The ImageWriter II printer driver also overrides the `GXRasterPackageBitmap` message. The `SD_PackageBitmap` override function fills in a buffer with the raster data, applying ImageWriter II run-length encoding and rotating the data. The `GXRasterPackageBitmap` message is described on page 4-100 in the chapter “Printing Messages.”

The `SD_RenderPage`, `SD_LineFeed`, and `SD_PackageBitmap` override functions are shown in the QuickDraw GX sample code.

## Terminating the Print Job

At the end of printing, your printer driver needs to clean up any storage that it has allocated. You can override the `GXShutDown` message to handle this task. The ImageWriter II printer driver overrides the `GXShutDown` message with the `SD_ShutDown` function, which is shown in Listing 3-18. The `GXShutDown` message is described on page 4-44 in the chapter “Printing Messages.”

---

**Listing 3-18** Terminating the print job

```
OSErr SD_ShutDown(void)
/*
    GXShutDown is called when the printing job is done. A good
    thing to do is to get rid of any additional storage.
*/
{
```

## Printer Drivers

```

/* clean up the globals */
SpecGlobalsHdl hGlobals = GetMessageHandlerInstanceContext();

/* get rid of the draft table if you allocated it */
DisposHandle((**hGlobals).draftTable);

/* get rid of allocated storage */
DisposHandle((Handle) hGlobals);

/* clear out the globals - to avoid double disposes */
SetMessageHandlerInstanceContext(nil);

return(noErr);
}

```

The `SD_ShutDown` function essentially reverses what the `SD_Initialize` function did at the start of printing, as shown in the section “Initializing the ImageWriter II Environment” beginning on page 3-24.

## Providing Compatibility in the ImageWriter II Driver

This section describes the override functions that comprise the portion of the ImageWriter II printer driver that provides compatibility with the Macintosh Printing Manager. This portion of the driver needs to override any of the Printing Manager compatibility messages that must take into account specific characteristics of the printing device. The functions that override such messages include the following:

- The `SD_ConvertPrintRecordTo` function uses knowledge of the ImageWriter printer characteristics to convert an old print record into a universal print structure.
- The `SD_ConvertPrintRecordFrom` function converts a universal print structure into Macintosh Printing Manager print record for the ImageWriter II.
- The `SD_PrintRecordToJob` function forwards the `GXPrintRecordToJob` message and then converts one of the resultant settings into an option in the job collection that is used by the driver.
- The `SD_PrValidate` function validates the current print record by filling it in with the current settings for the application and printing device.
- The `SD_PrJobInit` function first forwards the `GXPrJobInit` message to display the default settings in the Print dialog box, then disables some of the items, based on the current settings for the printing device.

All of the Macintosh Printing Manager compatibility messages, including those that correspond to these override functions, are described in the section “Compatibility Messages” beginning on page 4-147 in the chapter “Printing Messages.” The code for the overrides used by the ImageWriter II printer driver is found in the QuickDraw GX sample code.

## Color Printing

---

Printing in color is greatly simplified with QuickDraw GX. All of the color-processing information that you need to provide is specified in resources, which means that you don't need to override any messages or call any functions for color printing to work properly.

For raster printing devices, you specify color information in the raster driver preferences ('rdip') resource, which defines the color space, color profile, and other color information for each color plane on the printing device. For PostScript printing devices, you define the PostScript color space in the PostScript preferences ('pdip') resource. Both of these resources are described in the chapter "Printer Resources" in this book.

If you want to modify the color information at run time, you can override the `GXSetupImageData` message and modify the image data that QuickDraw GX created for your driver. The ImageWriter II printer driver uses the override function `SD_SetupImageData` to modify the halftone information when printing in graphics mode, as shown in Listing 3-11 on page 3-36. The `GXSetupImageData` message is described on page 4-92 in the chapter "Printing Messages." The data structures used to represent the imaging data for each imaging system are described in the chapter "Printing Messages" in this book.

For more information about color printing on PostScript printers, see the *PostScript Language Reference Manual*, 2nd Edition.

## Color Matching

---

Macintosh system software includes ColorSync, which provides color matching for drivers and applications. Color matching is the process that allows different devices to display the same color, providing the user with an accurate color representation in a device-independent manner.

You can provide color matching in your QuickDraw GX printer drivers without overriding any messages or calling any functions. All that you need to do is provide at least one color profile ('prof') resource in your driver. These resources, each of which describes how colors are represented on a specific device, are described in *Inside Macintosh: Advanced Color Imaging*. Although many printer drivers need only one color profile resource, you can define more than one such resource for your driver. Each profile can be associated with a specific page format or paper type. For example, the Apple Color Printer driver defines color profiles for three different media types that affect the appearance of color: coated paper, transparency film, and plain paper.

When you define a color profile resource for your printer driver, QuickDraw GX reads the resource data, creates a color profile object from the data, and associates that object with your printing device. If you want to create a color profile dynamically, you can override the `GXFetchTaggedData` message, which QuickDraw GX uses to read the



resource data. The `GXFetchTaggedData` message is described on page 4-45 in the chapter “Printing Messages.”

An application program can call the `GXFindPrinterProfile` function to query your printer driver. When an application calls this function, QuickDraw GX sends the corresponding `GXFindPrinterProfile` message. You can override this message to return to the application which profile you want to use. The application can then call the `GXSetPrinterProfile` function to change this information. Once again, QuickDraw GX sends the corresponding message—in this case, the `GXSetPrinterProfile` message. You can also override this message in your printer driver.

An application program can also call the `GXFindFormatProfile` function to query your printer driver to find out which color profile is used for a specific page format. When an application calls this function, QuickDraw GX sends the corresponding `GXFindFormatProfile` message. You can override this message to return to the application which profile you want to use. The application can then call the `GXSetFormatProfile` function to change this information. Once again, QuickDraw GX sends the corresponding message—in this case, the `GXSetFormatProfile` message. You can also override this message in your printer driver.

The `GXFindPrinterProfile`, `GXSetPrinterProfile`, `GXFindFormatProfile`, and `GXSetFormatProfile` message are described in the section “Color Profile Messages” beginning on page 4-62. The corresponding functions that cause these messages to be sent are described in *Inside Macintosh: QuickDraw GX Printing*.

If the application is changing color profiles on a per-format basis, you also need to override the `GXImagePage` message. One of the parameters for this message is a handle to imaging-system-specific data, which contains a handle to a color profile. In your override, you need to change which color profile the handle points to and forward message. The `GXImagePage` message is described on page 4-94 in the chapter “Printing Messages.”

## Color Matching on PostScript Devices

If you are writing a printer driver for a PostScript device, you need to perform a few additional tasks to optimize color matching:

- Choose the color space your driver uses. The color space tells QuickDraw GX what kind of PostScript operators to use when specifying colors for your printing device.
- If your PostScript device supports Level 2 of the PostScript language, offload the color-matching work to the printing device rather than having the Macintosh perform the matching work (which can be time consuming).
- Generate portable PostScript code to guarantee that the best output results are generated for any PostScript device with which your driver is used.

The following sections discuss these three tasks.

## Choosing a Color Space for a PostScript Printing Device

You must choose one of three color-space values for a PostScript device, each of which defines which PostScript operators QuickDraw GX uses to send color information. Table 3-8 shows the color-space values that you can use.

**Table 3-8** PostScript color-space choices

Color space	PostScript operators used by QuickDraw GX
gxRGBSpace	setrgbcolor and colorimage
gxCMYKSpace	setcmymcolor and colorimage
gxGraySpace	setgray and image

The color-space values are described the chapter “Colors and Color-Related Objects” in *Inside Macintosh: QuickDraw GX Objects*. If you specify the wrong color space, then PostScript errors are generated. For example, if you specify the `gxCMYKSpace` color-space and your driver is connected to a printer that does not support the `setcmymcolor` operator, an error occurs. You can get around this situation by generating portable PostScript code, as described in the section “Generating Portable PostScript” on page 3-53.

## Using PostScript’s Color Matching

If your printer driver is communicating with a PostScript device that supports Level 2 of the PostScript language, you can offload the color matching to the device rather than performing this time-consuming work in your driver. To do this, you need to specify two values in the PostScript preferences (‘pdip’) resource for your driver: the language-level field must be set to 2, and you must include `gxUseLevel2ColorOption` in the render options field of the resource.

When you specify these values in your PostScript preferences resource, QuickDraw GX generates PostScript code that is optimized for the PostScript Level 2 interpreter. This means that QuickDraw GX allows the PostScript interpreter to perform the color matching rather than calling `ColorSync` to do so. QuickDraw GX converts the color space and color profile of the objects being printed into a Level 2 color-space dictionary by using the `setcolorspace` operator. The colors for the objects are set with the `setcolor` operator, and bitmaps are drawn with the Level 2 dictionary form of the `image` operator. If a bitmap’s color space is defined as 5 or 8 bits per component, the `image` operator is used at 8 bits per component. If a bitmap’s color space is defined as 10 bits per component, the `image` operator is used at 12 bits per component.

Since not all QuickDraw GX color spaces cannot be translated to PostScript Level 2, QuickDraw GX might need to convert the color space of the object. For example, if the object specifies `gxCIESpace`, which cannot be emulated with the `setcolorspace` operator, QuickDraw GX converts the colors into `gxXYZSpace`, which can be emulated.

If you set the language level to 2 but do not specify `gxUseLevel2ColorOption` as a rendering option in your PostScript preferences resource, QuickDraw GX generates code that is optimized for the Level 2 interpreter, but calls `ColorSync` to perform color matching.

### Generating Portable PostScript

Since your PostScript driver can be used with a variety of printing devices, you need to be careful about which color space you specify. If you specify a color space that is not supported by the printing device, PostScript errors are generated. The best way to avoid this problem is to tell QuickDraw GX to generate portable PostScript code, which can be executed on any PostScript printing device and gives the best results that the printer can produce.

The only color space that is guaranteed to work on all PostScript devices is `gxGraySpace`. However, this color space generates grayscale even on a printer that supports color. To get QuickDraw GX to produce PostScript data that contains all of the color information but still renders on a monochrome device in grayscale, you need to specify two values in the PostScript preferences ('`pdip`') resource: include `gxPortablePostScriptOption` in the render options field, and specify `gxRGBSpace` as the device's color-space value.

When you specify these values, QuickDraw GX defines PostScript procedures that emulate any color operators that are not available on the output device. QuickDraw GX also generates PostScript code to set up a Level 2 color space that is based on the color profile defined for your driver. If your driver is connected to a device that has Level 2 specified in its profile, color-matched output is generated.

## Using Resources in Drivers

You need to provide a number of resources for drivers, just as you do for printing extensions. Some of these resources are required for all drivers, others are required for drivers that use a certain imaging system, and others are optional. All of the resources are described in the chapter "Printing Resources" in this book. Table 3-9 summarizes the resources that you can use in all printer drivers.

**Table 3-9** Resource types used to define a printer driver

Resource type	Count	Description
'over'	1, 2, or 3	Defines which messages your driver needs to receive
'isys'	1	Specifies what kind of imaging system the driver uses
'vers'	1 or more	Specifies which version of QuickDraw GX the driver uses

*continued*

**Table 3-9** Resource types used to define a printer driver (continued)

Resource type	Count	Description
'comm'	1 or more	Specifies printing-device communications parameters
'cust'	0 or 1	Specifies mappings from the Macintosh Printing Manager to the new driver architecture
'resl'	0 or 1	Specifies the horizontal and vertical resolution supported by a driver
'PREC'	0 or 1	Defines a default print record for your driver
'iobm'	0 or 1	Defines the timeout and buffering communications parameters for your driver
'cpts'	0 or 1	Specifies how your printing device can be removed and replaced on the network
'look'	1	Defines the type of communications that is used by the driver
'stat'	0 or more	Defines status messages for display
'plrt'	0 or more	Defines alert messages for display
'ppnl'	0 or more	Defines the contents of a panel that you are adding to a dialog box
'xdtl'	0 or more	Provides information that QuickDraw GX needs to execute panel controls
'ptyp'	0 or more	Defines characteristics of a paper type
'FREF'	1 or more	Specifies a file reference for the printer driver icons
'PACK'	1	Specifies Chooser package information
'LDEF'	1	Specifies Chooser list definition information
'BNDL'	1	Associates the printer driver with its icons and any files that it uses
'DLOG'	0 or more	Provides a dialog box template
'DITL'	0 or more	Contains an item list for a dialog box
'dctl'	0 or more	Specifies the dialog box control information for compatibility with Macintosh Printing Manager application dialog boxes
'stab'	0 or more	Defines a range of scaling for the scaling choice in the Print dialog box that is used for Macintosh Printing Manager compatibility
'ICN#'	1 or more	Defines a large (32-by-32 pixel) icon, 1-bit depth, with mask (if not provided, a generic driver icon is used)
'icl4'	0 or more	Defines a large icon, 4-bit depth (strongly recommended but not required)

**Table 3-9** Resource types used to define a printer driver (continued)

Resource type	Count	Description
'icl8'	0 or more	Defines a large icon, 8-bit depth (strongly recommended but not required)
'ics#'	0 or more	Defines a small (16-by-16 pixel) icon, 1-bit depth, with mask (if not provided, a generic driver icon is used)
'ics4'	0 or more	Defines a small icon, 4-bit depth (strongly recommended but not required)
'ics8'	0 or more	Defines a small icon, 8-bit depth (strongly recommended but not required)

This section provides examples of several of these resources as used in the ImageWriter II printer driver. The contents of the `oldapp.r` and `newapp.r` files, which contain the resource definitions for the ImageWriter II driver, are shown in the QuickDraw GX sample code.

You use some of the resources in Table 3-9 for user interface features, including the icons for your driver, resources that allow users to interact with your driver, and resources that the Chooser uses when the user selects your driver. The Finder interface resources, which are common to all Macintosh drivers, are described in *Inside Macintosh: More Macintosh Toolbox*. The icons that you must define for your driver when the user chooses it as the desktop printer are described in the section “Defining Desktop Printer Icons for Your Printer Driver” beginning on page 3-66.

All of the resources that you define for your printer drivers need to be loaded into the system heap and need to be purgeable. System resources are stored in the system heap as opposed to the application heap, where application resources are stored. Purgeable resources can be purged by the Memory Manager when space is required, as described in *Inside Macintosh: Memory*. You need to specify these attributes in the first line of every resource that you define for your driver, as is done in every resource example in this chapter.

### Defining Code Segments in Your Driver

The code segments that you use to implement your printer driver must use segment type `'pdvr'`, which is defined by the constant `gxPrinterDriverType`. The ID of your first code segment needs to be 0, and you need to increment the ID by 1 for each subsequent segment in your driver.

### Defining Version Compatibility for Your Printer Driver

Your printer driver must contain at least one version (`'vers'`) resource that defines its compatibility with QuickDraw GX. Version resources are used to record version information for Macintosh applications. You need to include a version resource with an ID of `gxPrintingDriverBaseID` that defines with which version of QuickDraw GX

## Printer Drivers

your driver is compatible. For the current version, the value of the first byte of the resource definition must be either 1 or 0. Listing 3-19 shows the version resource that defines QuickDraw GX compatibility for the ImageWriter II printer driver.

---

**Listing 3-19** The QuickDraw GX version resource for the ImageWriter II printer driver

```
resource 'vers' (gxPrintingDriverBaseID, sysHeap, purgeable)
{
    0x01, 0x00, release, 0x00,
    verUS,
    "1.00",
    "1.00, Copyright \251 Apple Computer, Inc. 1989-1993"
};
```

You can also include standard version resources in the resource files for your printer driver. These resources are described in *Inside Macintosh: Macintosh Toolbox Essentials*.

## Specifying Which Messages Your Driver Overrides

---

You must include an override ('over') resource in your printer driver to provide QuickDraw GX with a list of the printing messages that your driver is overriding. Each entry in the override resource specifies a message and information about the resource in which the code segment for the message override is found. You need to include separate override resources for universal messages and for messages specific to an imaging system. The override resource is described on page 6-13 in the chapter "Printing Resources."

The code segment information for each message includes the resource ID of the code segment and the offset into the code segment that contains the instruction to jump to the message-override function. The first 4 bytes of the code segment are reserved for use by QuickDraw GX and must be 0; thus, the first offset location is 4.

Each override resource in a printer driver has an ID of `gxPrintingDriverBaseID` for universal messages, an ID of `gxPrintingDriverBaseID+1` for messages specific to an imaging system, and an ID of `gxPrintingDriverBaseID+2` for Macintosh Printing Manager compatibility messages, as shown in Listing 3-20.

---

**Listing 3-20** Override resources for the ImageWriter II printer driver

```
#define firstOffset 4
#define segmentID NewSegID

resource gxOverrideType (gxPrintingDriverBaseID+0, sysHeap,
                        purgeable)
{
```

## Printer Drivers

```

    {
        gxInitialize,          segmentID, firstOffset + 0,
        gxShutDown,            segmentID, firstOffset + 4,
        gxDefaultPrinter,      segmentID, firstOffset + 8,
        gxDefaultFormat,       segmentID, firstOffset + 12,
        gxDefaultJob,          segmentID, firstOffset + 16,
        gxJobDefaultFormatDialog, segmentID, firstOffset + 20,
        gxJobFormatModeQuery,  segmentID, firstOffset + 24,
        gxRenderPage,          segmentID, firstOffset + 28,
        gxOpenConnection,      segmentID, firstOffset + 32,
        gxCloseConnection,     segmentID, firstOffset + 36,
        gxStartSendPage,       segmentID, firstOffset + 40,
        gxSetupImageData,      segmentID, firstOffset + 44,
    };
};

resource gxOverrideType (gxPrintingDriverBaseID + 1, sysHeap,
                        purgeable)
{
    {
        gxRasterPackageBitmap, segmentID, firstOffset + 48,
        gxRasterLineFeed,     segmentID, firstOffset + 52,
    };
};

#define segmentID      OldSegID

resource gxOverrideType (gxPrintingDriverBaseID + 2, sysHeap,
                        purgeable)
{
    {
        gxConvertPrintRecordTo, segmentID, firstOffset + 0,
        gxConvertPrintRecordFrom, segmentID, firstOffset + 4,
        gxPrintRecordToJob,      segmentID, firstOffset + 8,
        gxPrValidate,            segmentID, firstOffset + 12,
        gxPrJobInit,             segmentID, firstOffset + 16,
    };
};

```

Each resource in Listing 3-20 lists the messages that the ImageWriter II driver overrides. The first resource lists the universal messages, the second resource lists the messages specific to the raster imaging system, and the third resource lists the Macintosh Printing Manager compatibility messages. The code for the compatibility message overrides is located in a different code segment.

## Printer Drivers

Each message listed in the resources specifies the name of the message, the ID of that segment, and where to find the message in the jump table. The ID of the code segment for all of these messages is defined by the constant `segmentID`. The jump instructions to the code that implements the overrides are each 4 bytes long. The first jump is found at the first offset location (byte 4), and each subsequent jump is found 4 bytes beyond the previous one.

## Defining the Imaging System Type of Your Driver

You must include an imaging system (`'isys'`) resource in your printer driver to tell QuickDraw GX which imaging system your driver uses. The imaging system resource is described on page 6-33 in the chapter “Printing Resources.” You must include exactly one of these resources, and it must contain one of the imaging system constants that are shown in Table 3-10.

**Table 3-10** Imaging system values

Constant	Explanation
<code>gxRasterPrinterType</code>	The driver works with raster printing devices
<code>gxPostscriptPrinterType</code>	The driver works with PostScript printing devices
<code>gxVectorPrinterType</code>	The driver works with vector printing devices

The ImageWriter II printer is a raster printing device; thus, its imaging system resource specifies the `gxRasterPrinterType` value, as shown in Listing 3-21.

**Listing 3-21** The imaging system resource for the ImageWriter II printer driver

```
resource gxImagingSystemSelectorType (gxImagingSystemSelectorID,
                                     sysHeap, purgeable)
{
    gxRasterPrinterType
};
```

## Specifying How Your Driver Communicates With the Device

You need to specify how your printer driver communicates with the physical printing device that it is driving. You do this with two resource types: the communications (`'comm'`) resource and the buffering and input/output preferences (`'iobm'`) resource, known simply as the buffering resource. Each communications resource specifies the parameters for a particular kind of printing-device communications, as described in the section “The Communications (`'comm'`) Resource” beginning on page 6-36 in the chapter



“Printing Resources.” The buffering resource, which is described on page 6-61 in the chapter “Printing Resources,” specifies buffer size and timing parameters.

There are different kinds of communications resources—one for each kind of printing-device communications that QuickDraw GX supports. Each of these resource types has its own format. Listing 3-22 shows the communications resources that are defined for the ImageWriter II printer driver, which supports PAP, serial, and PrinterShare communications connections.

**Listing 3-22** Communications resources for the ImageWriter II printer driver

```
resource 'comm' (-4096, sysHeap, purgeable)
{
    PAP
    {
        1,          /* flow quantum */
        "",         /* AppleTalk address (filled in by Chooser) */
        0, 0, 0, 0
    };
};

resource 'comm' (-4095, sysHeap, purgeable)
{
    Serial
    {
        baud9600,    /* output baud rate */
        noParity,    /* output parity */
        oneStop,     /* output stop bits */
        data8,       /* output data size */
        0x00010000,  /* output handshaking */
        0x00000000,
        baud9600,    /* input baud rate */
        noParity,    /* input parity */
        oneStop,     /* input stop bits */
        data8,       /* input data bits */
        0,           /* input handshaking */
        0,
        1024,        /* input buffer size */
        ".AIn",      /* input driver name; filled in by Chooser */
        ".AOut"      /* output driver name; filled in by Chooser */
    };
};
```

## Printer Drivers

```
resource 'comm' (-4094, sysHeap, purgeable)
{
    PrinterShare
    {
        "",          /* AppleTalk address, filled in by Chooser */
        0
    };
};
```

You use the buffering resource to control the size of the buffers that your driver uses to communicate with the printing device. You also specify timeout values for your communications requests in this resource. Listing 3-23 shows the buffering resource that the ImageWriter II printer driver uses.

---

**Listing 3-23** The buffering and input/output preferences resource for the ImageWriter II printer driver

```
resource gxUniversalIOPrefsType (gxUniversalIOPrefsID, sysHeap,
                                purgeable)
{
    standardIO,
    4,          /* 4 buffers */
    2048,       /* 2 KB each (enough for 1 scan line of data) */
    10,         /* up to 10 I/O operations pending */
    1200,       /* open/close timeout of 1200 clock ticks */
    1200        /* read/write timeout of 1200 clock ticks */
};
```

## Defining Network Characteristics for Your Driver

---

You use capture ('cpts') resources to define network strings for your driver if your printing device supports network capture and removal. These resources are only needed if you are using QuickDraw GX's default implementation of the Printer Access Protocol (PAP) communications interface.

If you are using the default PAP implementation, you define four capture strings, each of which is used by QuickDraw GX to manage the capture and removal of your printing device. The capture strings can contain special substitution strings, as described in the section "The Capture ('cpts') Resource" beginning on page 6-63 in the chapter "Printing Resources."

Listing 3-24 shows the capture resources for the ImageWriter II printer driver.

**Listing 3-24** Capture resources for the ImageWriter II printer driver

```

resource gxCaptureType (gxCapturedAppleTalkType, sysHeap,
                        purgeable)
{
    "\0D011ImageShared"
};

resource gxCaptureType (gxUncapturedAppleTalkType, sysHeap,
                        purgeable)
{
    "\0D011ImageWriter"
};

resource gxCaptureType (gxCaptureStringID, sysHeap, purgeable)
{
    "\0X1B\0X62NAMELENPRINTERNAMETYPEPELENPRINTERTYPE\0X01*"
};

resource gxCaptureType (gxReleaseStringID, sysHeap, purgeable)
{
    "\0X1B\0X62NAMELENPRINTERNAMETYPEPELENPRINTERTYPE\0X01*"
};

```

## Defining Status Messages

You can define your own status ('stat') resources for sending informational messages to the desktop printer window while a document is printing. Each status resource includes a status-type indicator, a status ID value, an alert value, and the status string. Listing 3-25 shows a status resource that defines the status text string displayed while the ImageWriter II printer driver is sending document data to the printer.

**Listing 3-25** A status resource for the ImageWriter II printer driver

```

resource 'stat' (kDriverStatus, sysHeap, purgeable)
{
    'IWII',

    {
        gxInformationalStatus, 1, 0, "Sending data to printer";
        gxUserAlert, 1, kDriverStatus, "Please check that the printer
is on-line";
    }
};

```

## Resources for Compatibility With the Macintosh Printing Manager

You can customize the way that your driver supports Macintosh Printing Manager printing by defining a customization ('cust') resource. QuickDraw GX provides a default version, so this an optional resource.

The customization resource defines the resolution, translator settings for the driver, the pattern stretch factor, and how it translates Macintosh Printing Manager calls into QuickDraw GX printing messages. You can also fine tune the handling of certain geometries by the QuickDraw GX translator. The customization resource is described on page 6-47. The program that translates Macintosh Printing Manager calls to QuickDraw GX messages is described in *Inside Macintosh: QuickDraw GX Environment and Utilities*.

The customization resource from the ImageWriter II printer driver is shown in Listing 3-26.

**Listing 3-26** The customization resource for the ImageWriter II printer driver

```
resource gxCustType (gxCustID, sysHeap, purgeable)
{
    144,144,                /* 300 dpi device */
    defaultUpDriver,        /* use LaserWriter interface */
    {2,2},                  /* pattern stretch of 2 */
    gxOptimizedTranslation /* use default translator settings */
}
```

## Defining Device Characteristics Specific to an Imaging System

Most of the unique qualities and characteristics of the printing device for which you are implementing a driver are described in the resources that you provide. The ImageWriter II printer driver includes three resources for the raster imaging system. You need to include resource definitions for the PostScript imaging system if you are developing a driver for a PostScript printing device. You need to include resource definitions for the vector imaging system if you are developing a driver for a vector printing device. All of these resources are described in the chapter “Printing Resources” in this book.

You must include a raster preferences ('rdip') resource for a raster printer driver. This resource specifies imaging options for your driver, including color information. The raster preferences resource for the ImageWriter II printer driver, which is shown in Listing 3-27, includes color information for each color plane provided by the printing device.

**Listing 3-27** The raster preferences resource for the ImageWriter II printer driver

```

resource gxRasterPrefsType (gxRasterPrefsID, sysHeap, purgeable)
{
    gxDefaultRaster,          /* default options are fine */

    0x00900000,0x00900000,    /* 144X144 dpi device */
    16,                      /* min band size == 2 head heights */
    0,                      /* max band size (0 is full page) */
    0x00004000,              /* RAM percentage (25%) */
    100*1024,                /* RAM slop (100K) */
    4,                      /* 4-bit device */
    {
        /* dithering offscreen */
        3,
        gxDontSetHalftone + gxDotTypeIsDitherLevel,
        0x002D0000,          /* angle unused for dithering */
        0x003C0000,          /* freq unused for dithering */
        4,                  /* dithering with level of 4 */
        gxLuminanceTint,     /* tint space unused for dithering */
                                /* dot color & background unused */
        gxRGBSpace, gxNoProfile, 0, 0, 0, 0,
        gxRGBSpace, gxNoProfile, 0xFFFF, 0xFFFF, 0xFFFF, 0,
        gxRGBSpace,          /* halftone space */
        gxIndexedSpace,      /* indexed color space */
        gxPrintingDriverBaseID, /* the color set to use */
        1 /* the color profile to use */
    };
};

```

The raster package ('rpck') resource controls how bitmap data is packed into rasters for your driver. QuickDraw GX provides a default version of this resource, so providing your own version is optional. Listing 3-28 shows the raster package resource for the ImageWriter II driver.

**Listing 3-28** The raster package resource for the ImageWriter II printer driver

```

resource gxRasterPackType (gxRasterPackID, sysHeap, purgeable)
{
    /*
        The packing buffer size. For the ImageWriter II, this
        is the # of bytes in the largest single packaged line.
    */
    2500,

```

## Printer Drivers

```

4,          /* this is CMYK (so colorsPasses == 4) */
16,         /* print head is 16 pixels high */
2,          /* it takes 2 passes to achieve the 16 pixels */
1,          /* there is a 1 pixel difference between
           these two passes */
gxInterlaceColor, /* avoid ribbon contamination */
};

```

The final raster resource provided by the ImageWriter II driver is the raster package controls ('ropt') resource, which you use to define how some forms of line feeding are performed on your printing device. Listing 3-29 shows the ImageWriter II printer driver version of this resource.

---

**Listing 3-29** The raster package controls resource for the ImageWriter II printer driver

```

resource gxRasterPackOptionsType (gxRasterPackOptionsID, sysHeap,
                                purgeable)
{
    gxPrintingBaseID,
    gxPrintingBaseID + 10,

    /* forward line-feed characteristics */
    98,          /* max line-feed amount is 98 */
    gxRasterNumToASCII, /* express line-feed as ASCII */
    2,          /* minimum width is 2 */
    "0",        /* and pad with zeros */
    "\0X1BT",   /* <esc>T == set line-feed size */
    "\0X1Bf\0X0A", /* <esc>f<lf> == direction forward,
                   do line feed */

    /* reverse line-feed characteristics */
    98,          /* max line-feed amount is 98 */
    gxRasterNumToASCII, /* express line-feed as ASCII */
    2,          /* minimum width is 2 */
    "0",        /* and pad with zeros */
    "\0X1BT",   /* <esc>T == set line-feed size */
    "\0X1Br\0X0A", /* <esc>r<lf> == direction reverse,
                   do line feed */
};

```

## User Interface and Chooser Support

---

You need to provide at least a minimal user interface to your printer driver, including support for the user to select printing choices, an interface to the Chooser for selecting your printer driver, and a collection of icons to represent your printer driver on the user's screen.

The contents of the `ChooserSupport.r` file, which contains the Chooser resource definitions for the ImageWriter II driver, are shown in the QuickDraw GX sample code. The icon definitions for the ImageWriter II driver are found in the `newapp.r` resource file, which is also shown in the QuickDraw GX sample code.

### Providing Printing Choices

---

You can use the dialog panel (`'ppnl'`) resources to provide QuickDraw GX with information to be displayed in dialog panels.

- The dialog panel resource gives QuickDraw GX the information necessary to display a panel.
- The extended dialog panel resource (resource type `'xdtl'`) provides QuickDraw GX with the information it needs to execute panel controls.

These resources are described in *Inside Macintosh: QuickDraw GX Printing*.

You can also modify the choices that are displayed to the user in the print dialog boxes by changing the values in the job collection. You can access and modify these values by using the appropriate tags (as described in the section “Using the Printing-Related Collections” beginning on page 3-20). Among the items that you might need to modify are the following:

- paper-tray availability
- print-quality choices
- print-scaling boundaries
- printing-file options

### The Chooser and Your Driver

---

You need to define resources for your printer driver to provide a Chooser interface, which allows the user to select your driver from among the printer drivers available on the computer.

The Chooser uses the look (`'look'`) resource to generate the pop-up menu list of “Connect via:” options that it displays when the user selects your driver. Listing 3-30 shows an example of a look resource for the ImageWriter II printer driver, which supports serial, PAP, and PrinterShare connections.

**Listing 3-30** The look resource for the ImageWriter II printer driver

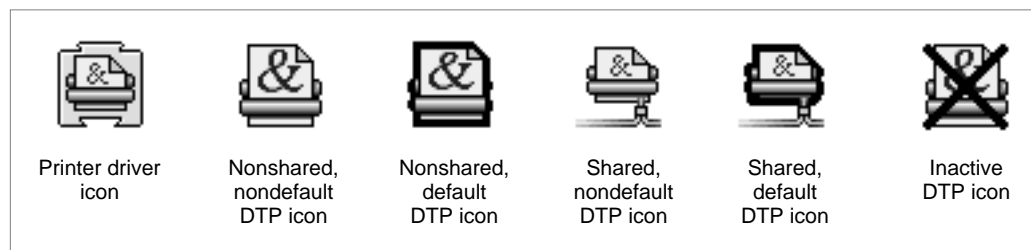
```
resource 'look' (-4096, sysHeap, purgeable)
{
    2,      /* use the second in the list as the default */
    {
        "AppleTalk",    -4096,    isAppleTalk,    "ImageWriter";
        "Serial",        -4095,    iconCells,        "Modem Port";
        "Servers",        -4094,    isAppleTalk+isPrinterShare,
                                "ImageWriterIIIS";
    };
};
```

The look resource accesses each communications resource that you specify for your driver to determine parameter values for the type of connection that the user selects.

## Defining Desktop Printer Icons for Your Printer Driver

You need to provide six icons for your printer driver: one to represent your driver in the Extensions folder inside of the System folder, and five others to represent different states of the desktop printer. Figure 3-3 shows the six icons for the Apple LaserWriter driver.

QuickDraw GX automatically imposes certain status icons over the desktop printer (DTP) icons to indicate certain conditions to the user. The status icons are described in the next section.

**Figure 3-3** The Apple LaserWriter printer driver icons



## Printer Drivers

QuickDraw GX uses the printer driver icon to represent the driver in the Extensions folder in the System folder. The desktop printer icons are used to represent different states of the driver when the user has made it a desktop printer.

- The nonshared, nondefault desktop printer icon is displayed on the desktop when the printer is not the default printer and is not a shared printer. This icon is also used to represent your printer driver in the Chooser.
- The nonshared, default desktop printer icon is displayed on the desktop when the user has selected a nonshared printer as the default printer.
- The shared, nondefault desktop printer icon is displayed on the desktop of both the client and server computers when a shared printer is not the default printer.
- The shared, default desktop printer icon is displayed on the desktop of both the client and server computers when a user has selected a shared printer as the default printer.
- The inactive desktop printer icon is displayed when a desktop printer is not on the startup disk desktop or when QuickDraw GX is not active (for example, if the user has installed QuickDraw GX but has booted the computer with extensions turned off).

You need to follow certain guidelines when designing your icons:

- Each default desktop printer icon must resemble the icon for the corresponding nondefault desktop printer icon and must have a bold (3 pixels wide) outline drawn around it.
- Each shared desktop printer icon must resemble the printing device with networking cables slightly overlapping at the bottom of the printing device.
- The inactive desktop printer icon must resemble the nonshared, nondefault desktop printer icon and must have bold (3 pixels wide) crossing lines drawn over it.

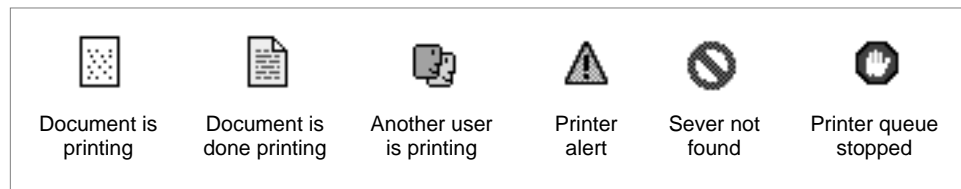
One strategy for designing your icons is to design your nondefault desktop printer icons with room for the border that is added for their “default” versions. This simplifies your design process and provides the user with icons that have a consistent appearance and are not distorted.

You need to carefully place the network cables in your shared desktop printer icons because of how they appear when the status icons are overlaid on them. The status icons are more noticeable when the cables are up one pixel from the edge and when the “Y” area of the cables are located in the right half of the icon space. The status icons are described in the next section.

## The Desktop Printer Status Icons

QuickDraw GX uses the desktop-printer status icons to convey additional printing information to the user. These are stand-alone icons that are imposed on the lower-left quadrant of the desktop printer icons whenever the status of a desktop printer changes. Each of the status icons is designed with unique color characteristics to help visually distinguish them. Figure 3-4 shows the QuickDraw GX desktop-printer status icons.

**Figure 3-4** The QuickDraw GX desktop-printer status icons































When a desktop printer is available but not currently being used, the desktop printer icon is displayed without any status added. QuickDraw GX automatically adds the status icon whenever the printer's status changes, as follows:

- When one of the user's documents is printing on the desktop printer, the "document is printing" status icon is overlaid on the desktop printer icon.
- When one of the user's documents has finished printing on the desktop printer, the "document is done printing" status icon is overlaid on the desktop printer icon.
- When a shared desktop printer is printing a document that was sent to it by another user, the "another user is printing" status icon is overlaid on the desktop printer icon.
- When a desktop printer requires user attention for conditions such as paper jams or an empty paper tray, the "printer alert" status icon is overlaid on the desktop printer icon.
- When the desktop printer server cannot be found on the network, the "server not found" status icon is overlaid on the desktop printer icon.
- When the user chooses the Stop Printer Queue item in the Printing menu, the "printer queue stopped" status icon is overlaid on the desktop printer icon.

You provide four desktop printer icons for your printer driver, and QuickDraw GX automatically overlays six different icons on them to convey status information. This means that a desktop printer can be represented on the user's desktop by 28 different icons, as shown in Figure 3-5.

**Figure 3-5** Desktop printer icons showing printer status

	Nonshared, nondefault	Nonshared, default	Shared, noncurrent	Shared, current
Printer is available				
Printer is printing your document				
Printer is done printing your document				
Printer is printing someone else's document				
Printer needs user's attention				
Server not found				
Printer queue has been stopped				

### Bundling Your Printer Driver Icons

You need to create a bundle ( ' BNDL ' ) resource for your printer driver, just as you do for any Macintosh application program. The bundle resource, which is described in *Inside Macintosh: Macintosh Toolbox Essentials*, associates your printer driver with its icons and with any files that it creates.

QuickDraw GX needs to map the local IDs of your desktop printer icons (which are described in the previous section) into specific IDs to properly use them. For this to happen, you must define a file reference ( ' FREF ' ) resource for each type of desktop

## Printer Drivers

printer icon that you include with your driver. The file types listed in Table 3-11 must be used as shown.

**Table 3-11** File types for desktop printer icons

Icon file type	Icon usage
'dpnn'	Nonshared, nondefault desktop printer
'dpcn'	Nonshared, default desktop printer
'dpns'	Shared, nondefault desktop printer
'dpcs'	Shared, default desktop printer
'dvcl'	Desktop printer when QuickDraw GX is not active
'dppz'	Printer driver in the Extensions folder when QuickDraw GX is active
'pdvr'	Printer driver in the Chooser when QuickDraw GX is active, and printer driver in the Extensions folder when QuickDraw GX is not active

The bundle resource for the ImageWriter II printer driver defines the local ID and file reference IDs for each of the desktop printer icon definitions that follow. The resource definition is shown in Listing 3-31.

**Listing 3-31** The bundle resource for the ImageWriter II printer driver

```
resource 'BNDL' (gxPrintingDriverBaseID + 1, sysHeap, purgeable)
{
    kCreatorType,
    0,
    {
        'ICN#', { 0, gxPrintingDriverBaseID + 2;
                  1, gxPrintingDriverBaseID + 3;
                  2, gxPrintingDriverBaseID + 4;
                  3, gxPrintingDriverBaseID + 5
                },
        'FREF', { 0, gxPrintingDriverBaseID + 2;
                  1, gxPrintingDriverBaseID + 3;
                  2, gxPrintingDriverBaseID + 4;
                  3, gxPrintingDriverBaseID + 5;
                  0, gxPrintingDriverBaseID + 1
                }
    }
};
```

The ImageWriter II printer driver includes a file reference resource for its file type signature and one file reference resource for each type of desktop printer icon, as shown in Listing 3-32.

**Listing 3-32** The file reference resources for the ImageWriter II printer driver

```
resource 'FREF' (gxPrintingDriverBaseID + 1, sysHeap, purgeable)
{ kFileType, 0, "" };

resource 'FREF' (gxPrintingDriverBaseID + 2, sysHeap, purgeable)
{ 'dpnn', 0, "" };

resource 'FREF' (gxPrintingDriverBaseID + 3, sysHeap, purgeable)
{ 'dpns', 1, "" };

resource 'FREF' (gxPrintingDriverBaseID + 4, sysHeap, purgeable)
{ 'dpcn', 2, "" };

resource 'FREF' (gxPrintingDriverBaseID + 5, sysHeap, purgeable)
{ 'dpcs', 3, "" };
```

The ImageWriter II printer driver defines icons in various sizes and resolution for display on the user's desktop. Note that each related icon (the various sizes and resolutions for each purpose) shares the same resource ID. For example, each of the icons used to represent a desktop printer that is shared and default ('dpcs') has resource ID `gxPrintingDriverBaseID + 5`, as shown in Listing 3-33. The actual data for the icons can be found in the QuickDraw GX sample code.

**Listing 3-33** The icon resources for the ImageWriter II printer driver

```
/*
    Icons in various sizes and resolutions for the different
    representations of the desktop printer are included here.
*/

/* nonshared, nondefault desktop printer icon definitions*/
resource 'ics#' (gxPrintingDriverBaseID + 2, sysHeap, purgeable)
{
    /* icon resource goes here */
};

resource 'ics4' (gxPrintingDriverBaseID + 2, sysHeap, purgeable)
{
    /* icon resource goes here */
};
```

## Printer Drivers

```

resource 'ics8' (gxPrintingDriverBaseID + 2, sysHeap, purgeable)
{
    /* icon resource goes here */
};

resource 'ICN#' (gxPrintingDriverBaseID + 2, sysHeap, purgeable)
{
    /* icon resource goes here */
};

resource 'icl4' (gxPrintingDriverBaseID + 2, sysHeap, purgeable)
{
    /* icon resource goes here */
};

resource 'icl8' (gxPrintingDriverBaseID + 2, sysHeap, purgeable)
{
    /* icon resource goes here */
};

    /* nondefault, shared desktop printer icon definitions*/
resource 'ICN#' (gxPrintingDriverBaseID + 3, sysHeap, purgeable)
{
    /* icon resource goes here */
};

resource 'icl4' (gxPrintingDriverBaseID + 3, sysHeap, purgeable)
{
    /* icon resource goes here */
};

resource 'icl8' (gxPrintingDriverBaseID + 3, sysHeap, purgeable)
{
    /* icon resource goes here */
};

    /* default, nonshared desktop printer icons */
resource 'ICN#' (gxPrintingDriverBaseID + 4, sysHeap, purgeable)
{
    /* icon resource goes here */
};

resource 'icl4' (gxPrintingDriverBaseID + 4, sysHeap, purgeable)
{
    /* icon resource goes here */
};

resource 'icl8' (gxPrintingDriverBaseID + 4, sysHeap, purgeable)
{
    /* icon resource goes here */
};

    /* default, shared desktop printer icon definitions*/
resource 'ICN#' (gxPrintingDriverBaseID + 5, sysHeap, purgeable)
{
    /* icon resource goes here */
};

```

## Printer Drivers

```
resource 'icl4' (gxPrintingDriverBaseID + 5, sysHeap, purgeable)
{
    /* icon resource goes here */
};

resource 'icl8' (gxPrintingDriverBaseID + 5, sysHeap, purgeable)
{
    /* icon resource goes here */
};
```

